UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Double Degree in Computer Science and Math

DEGREE WORK

Independence tests based on embeddings in functional spaces

Author: Roberto Alcover Couso
Advisor: Alberto Suárez González

junio 2019

Roberto Alcover Couso
Independence tests based on embeddings in functional spaces

Roberto Alcover Couso
C\ Francisco Tomás y Valiente Nº 11

PRINTED IN SPAIN

Agradecimientos

To my parents who raised me, my sister who made me and the friends who embraced me.

My tutor who through all my nervousness stayed and helped me with every detail of this project.

My Statistics teacher who showed me what a beautiful research field is.

Finally to the music, without it this last year would've been miserable, so many hours looking to the screen.

Long live to the walls we crashed through, we will be remembered.

[Another student]

Resumen

Uno de los problemas fundamentales en estadística es identificar dependencias entre variables aleatorias. Los test estándar de dependencia como el ρ de Pearson, no pueden identificar todas las posibles dependencias no lineales. La principal dificultad en el diseño de tests de independencia efectivos es la gran variedad de posibles patrones de asociación que pueden aparecer en los datos.

En este proyecto abordaremos este problema usando tres enfoques diferentes:

La primera forma de abordarlo, usaremos *mean embedding* para mapear una distribución de probabilidad en un elemento de un espacio de Hilbert con núcleo reproductor (RKHS). Como estos espacios están dotados de una métrica, para producir un test de independencia solo se tiene que computar la distancia entre el elemento en el RKHS correspondiente a la distribución conjunta de las variables aleatorias consideradas, y al elemento al que el producto de las marginales se transforma. Dado que la distribución conjunta de dos variables aleatorias independientes es el producto de las marginales, si la distancia es distinta de cero, las variables son dependientes. Si el RKHS es lo suficientemente rico, un valor de cero en ésta distancia caracteriza a su vez independencia. Éste test de independencia se refiere a el criterio de independencia de Hilbert-Schmidt (HSIC).

En un segundo enfoque, usaremos proyecciones aleatorias no lineales para representar los datos en un espacio multidimensional. Siempre que se cumplan algunas condiciones matemáticas para las proyecciones no lineales, las correlaciones lineales en este espacio extendido de características aleatorias determinan dependencias no lineales en el espacio original. Haciendo uso de esta propiedad, es posible diseñar un test de independencia, conocido en la literatura como el coeficiente de dependencia aleatorizado (RDC).

Finalmente, un tercer enfoque, computando una distancia específica entre las distribuciones de probabilidad acumuladas de variables aleatorias. Probaremos como para dimensiones mayores que uno, esta distancia se vuelve más difícil de calcular. Por ello, esta cantidad será expresada en términos de una distancia entre las correspondientes funciones características, las cuales en general son más manejables. Éste método es referido como *Distance Covariance* (DCOV) test.

En este trabajo desarrollaremos un conjunto de herramientas de Python para implementar estas diferentes medidas de dependencia con el objetivo de poder implementar tests de independencia basados en ellas. Las propiedades de estos test son analizadas y comparadas. De ésta evaluación, concluimos que el test RDC es tanto más efectivo como eficiente en la mayor parte de los casos considerados.

Palabras clave

mean embedding, espacio de Hilbert con núcleo reproductor (RKHS), criterio de independencia de Hilbert-Schmidt (HSIC), coeficiente de dependencia aleatorizado (RDC), Distance Covariance (DCOV)

Abstract

One of the fundamental problems in Statistics is the identification of dependencies between random variables. Standard tests of dependence such as Pearson's ρ, cannot identify all possible non-linear dependences. The main difficulty in the design of effective tests of independence is the wide variety of association patterns that can be encountered in the data.

In this work we will address this problem using three different approaches:

In a first approach, *mean embedding* is used to map a probability distribution onto an element of a Reproducing Kernel Hilbert Space (RKHS). Since such space is endowed with a metric, to perform an independence test one simply needs to compute the distance between the element in the RKHS that corresponds to the joint distribution of the random variables considered, and the element onto which the product of the marginal is transformed. Given that the joint distribution of two independent random variables is the product of the marginals, if this distance is non-zero, the variables are dependent. If the RKHS is sufficiently rich, a zero value of this distance also characterizes independence. This independence test is referred to as the Hilbert-Schmidt Independence Criterion (HSIC).

In a second approach, random non-linear projections are used to represent the data in a high-dimensional space. Provided that some mathematical conditions are fulfilled for the non-linear projections, linear correlations in this extended space of random features characterize non-linear dependencies in the original one. Taking advantage of this property, it is possible to design a test for independence, which is known in the literature as the Randomized Dependence Coefficient (RDC) test.

Finally, in a third approach, one computes a specific distance between the cumulative distribution functions of the random variables. We will show how for dimensions greater than one, this distance becomes more difficult to compute. Therefore, this quantity is expressed in terms of a distance between the corresponding characteristic functions, which are in general more manageable. This method is referred to as the *Distance Covariance* (DCOV) test.

In this work, a set of Python tools have been developed to compute these different dependence measures, so that independence tests based on them can be carried out. The properties of these tests are the analyzed and compared in an exhaustive set of experiments. From this evaluation, we conclude that the RDC test is both effective and efficient in most of the cases considered.

Keywords

mean embedding, Reproducing Kernel Hilbert Space (RKHS), Hilbert-Schmidt Independence Criterion (HSIC), Randomized Dependence Coefficient (RDC), Distance Covariance (DCOV)

Table of Contents

Lists

List of codes

List of equations

List of figures

1

Introduction

Since the 8th century, when Al-Khali(717-786) wrote the *Book of Cryptographic Messages* which contains the first use of permutations and combinations [1] humans have shown interest on the study of the likelihood of events. In the eighteenth century with Jacob Bernoulli's *Ars Conjectandi* (posthumous, 1713) [2] a version of the fundamental law of large numbers was proven, which states that in a large number of trials, the average of the outcomes is likely to be very close to the expected value, probability became one of the main mathematical fields, introducing probability measures. Probability measures are widely used in hypothesis testing, density estimation, Markov chain and Monte Carlo to give some examples. In this work our main focus will be hypothesis testing, mainly homogeneity testing.

The goal in homogeneity testing is to accept or reject the null hypothesis $\mathcal{H}_0$:$\mathbb{P} = \mathbb{Q}$, versus the alternative hypothesis $\mathcal{H}_1$:$\mathbb{P} \neq \mathbb{Q}$, for a class of probability distributions $\mathbb{P}$ and $\mathbb{Q}$. For this purpose we will define a metric γ such that testing the null hypothesis is equivalent to testing for $\gamma(\mathbb{P}\mathbb{Q}) = 0$. We are specially interested in testing for independence between random vectors, which is a particular case of homogeneity testing, using $\mathbb{P} = \mathbb{P}_{\mathcal{X}\mathcal{Y}}$ and $\mathbb{Q} = \mathbb{P}_{\mathcal{X}} \cdot \mathbb{P}_{\mathcal{Y}}$, where $\mathbb{P} = \mathbb{P}_{\mathcal{X}\mathcal{Y}}$ is the joint distribution and $\mathbb{Q} = \mathbb{P}_{\mathcal{X}} \cdot \mathbb{P}_{\mathcal{Y}}$ is the product of the marginal distribution.

Measuring the existence of dependence between variables is a classical, yet fundamental problem in statistics. Starting with Auguste Bravais and Francis Galton's correlation coefficient defined as a product-moment, and it's relation with linear regression *Stigler (1989)*, many techniques have been proposed, developed and studied. Nowadays this subject is of fundamental importance in scientific fields such as Physics, Chemistry, Biology, and Economics. A practical application is Principal Component Analysis (PCA), which is a statistical procedure that converts a set of observations of possibly correlated variables into a set of linearly uncorrelated variables called principal components.

The goal of this project is to study, implement in python and compare some state of the art dependence measures in order to analyze for which scenarios one measure may outperform others. As we go along through the document, we will present three main approaches of non-linear dependence measures will be presented: In a first approach, *mean embedding* is used to map a probability distribution onto an element of a Reproducing Kernel Hilbert Space (RKHS). Since such space is endowed with a metric, to perform an independence test one simply needs to compute the distance between

the element in the RKHS that corresponds to the joint distribution of the random variables considered, and the element onto which the product of the marginal is transformed. Given that the joint distribution of two independent random variables is the product of the marginals, if this distance is non-zero, the variables are dependent. If the RKHS is sufficiently rich, a zero value of this distance also characterizes independence. This independence test is referred to as the Hilbert-Schmidt Independence Criterion (HSIC) .

On the second one, which is an approximation of the *Hirschfeld-Gebelein-Rényi's Maximum Correlation Coefficient* (HGR), defined by Gebelein in 1941 [3]. For this coefficient, random non-linear projections are used to represent the data in a high-dimensional space. This random non-linear projections will be selected as the equivalent of a Gaussian Kernel: The non-linearity comes from a sine and cosine (being the development of an exponential), and the random aspect will come from multiplying our variables with a Normal sample. Linear correlations in this extended space of random features characterize non-linear dependencies in the original one. Taking advantage of this property, it is possible to design a test for independence, which is known in the literature as the Randomized Dependence Coefficient (RDC) test.

Finally our last approach will be by measuring the distance between their respective cumulative distribution functions. We will show how for dimensions greater than one this distance gets more complex to calculate, therefore we will express this distance using their respective characteristic functions, which always exist and are fairly easy to manage for any given dimension. This idea will be used to generate an homogeneity test called Energy distance, which will be used as a bridge to introduce the distance covariance criterion (DCOV) , which will consist on applying this defined metric between probability distributions to the joint distribution and the product of the marginals.

Our aim was for our analysis to be as complete as possible, nevertheless given the time constraints of an undergraduate thesis project, the complete scope of the project wasn't clearly delimited at the beginning. Therefore, for the development process, we followed an agile methodology, where the advance of the project was discussed during periodic weekly meetings.

Now we will present a brief structure of the work, which is as follows:

In the beginning of the work, which is composed of Chapter 2, we will present a brief introduction of the previous mathematical knowledge needed in order to understand completely the whole document, then we will explain in detail each of the dependence measures mentioned above, starting with HSIC, followed by DCOV and concluding with RDC.

With all the mathematical content explained, we will present the software aspect of the project. Starting with the design. Going in detail about the structure of the software, we will introduce this section with the analysis of requirements of the project which will serve as a connecting flow to transition to the design aspects of the project, talking about the relationship between the structures which will be implemented and how they interact with each other. This will lead us directly to talk about the development

aspect, where we will explain how the requirements were managed and problems which may have risen were solved. Then, we will explain a little bit about version control of the project, which tools were used and concluding with the software development methodology which was followed.

We will continue with the experiments which were performed in order to compare each of the dependence measures, where we will find that RDC is not only more efficient but generally more effective in general. To conclude, we will present a final discussion about the results obtained, where we will dive into specific details about when DCOV may be better than RDC, and talk about future work branches for this project.

2

State of The Art: Dependence Measures

On the grounds that this project is built with a really strong mathematical component, the following section is created in order to introduce the reader to the previous mathematical knowledge which will be useful and used during this whole chapter, being Hilbert Spaces, Feature maps, Kernels and Reproducing Kernel Hilbert Spaces (RKHS). Then, we will dive into explaining the concept of mean embedding and study an homogeneity test called maximum mean discrepancy using this concept, from this test we will go to our first dependence measure, Hilbert-Schmidt Independence Criterion (HSIC).

Secondly, we will study the concept of energy distance, which is a generalization of the concept of $\mathcal{L}^2$ distance between two CDFs. This, like in the previous section will help us to develop a homogeneity test, which will be used to create an independence test called distance covariance (DCOV).

Finally, we will conclude with the study of the Randomized Dependence Coefficient, which is an approximation of the Hirschfeld-Gebelein-Rényi's Maximum Correlation Coefficient [3]. This will be our last dependence measure, from all this independence criteria we will study their respective associated statistics and convergence.

2.1. Previous mathematical content

This section will introduce the concept of Hilbert Space, feature maps, kernel functions and RKHS, . This concepts will be the basic knowledge needed in order to understand the following 4 sections of this chapter, where Kernells and feature maps will be needed in order to construct HSIC and RDC which are two of the three dependence measures which this work revolvs around. Most content is taken from [4].

2.1.1. Hilbert Space

A Hilbert space, generalizes the notion of Euclidean space, this will allow us to extrapolate properties of finite dimensional spaces to infinite dimensional spaces, such as functional spaces. Hilbert spaces are important in various fields, for example in partial differential equations.

Definition 2.1.1. *Any metric space (Space induced by an inner product) which is complete (Any*

Cauchy sequence converges with respect of it's norm) is called a Hilbert space.

For more information about Hilbert Spaces head to [5] chapter 2.

2.1.2. Feature maps

Classical theory of statistics covers the problem linear dependences, however real world problems often contain nonlinear dependencies. One approach that one can take of the knowledge of linear dependences is to transform the data into a different space where nonlinear dependences are transformed into linear ones.

Definition 2.1.2. *Let $\mathcal{H}$ be a Hilbert space, called the feature space, X an input set and x a sample from the input set. A feature map is a map $\phi : X \to \mathcal{H}$ from inputs to vectors in the Hilbert space. The vectors $\phi(x) \in \mathcal{H}$ are called feature vectors.*

This spaces play an important role in machine learning, since they map any type of input data into a space with a well defined-metric. If the feature map is a nonlinear function it can change the relative position between data points like in 2.1 making classification much easier in the feature space.

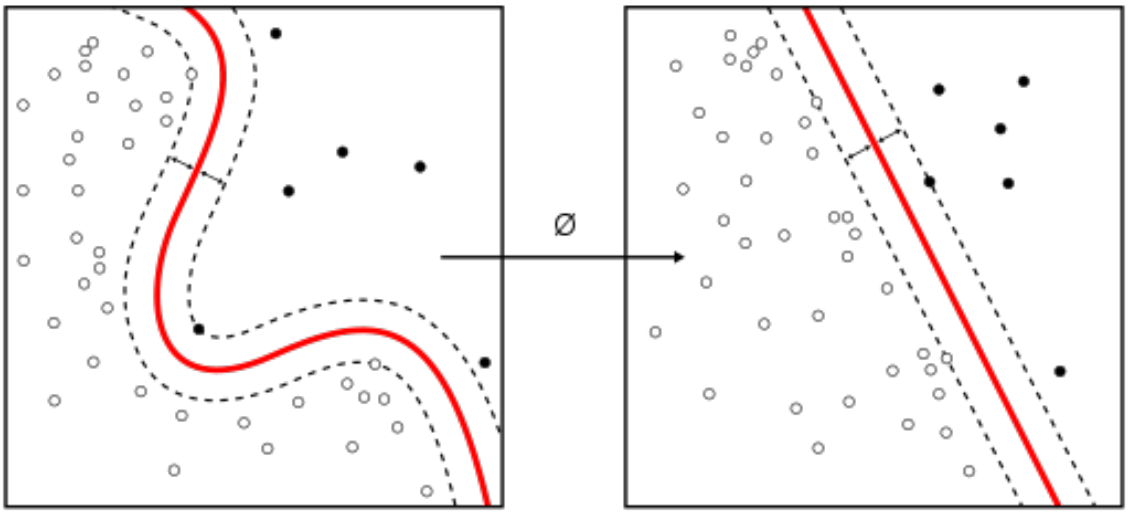


Figure 2.1: Figure taken from: Wikipedia Alisneaky, svg version by User:Zirguezi creativecommons.org/licenses/by-sa/4.0 . Which shows graphically how a feature map can modify a dataset.

By definition of the inner product, every feature map has an associated kernel.

Definition 2.1.3. *Let $\phi : X \to \mathcal{H}$ be a feature map. The inner product of two inputs mapped to feature space defines a kernel:*

$$k(x, x') =< \phi(x), \phi(x') >_{\mathcal{H}}$$

where $< \cdot, \cdot >_{\mathcal{H}}$ *is the inner product defined on* $\mathcal{H}$*.*

Demonstration can be found in [4] Theorem 1.

2.1.3. Reproducing kernel Hilbert spaces (RKHS)

Reproducing kernel Hilbert spaces (RKHS) were introduced with Moore's work (1916) and Aronszajin (1950). They are a special subset of Hilbert spaces that have a kernel with the reproducing property.

Definition 2.1.4. *A bivariate function k on a given space S is said to be a reproducing kernel for* $\mathcal{H}$ *if for every* $t \in S$*,* $k(\cdot, t) \in \mathcal{H}$ *and k satisfies the reproducing property, that for every* $f \in \mathcal{H}$ *and* $t \in S$ $f(t) =< f, k(\cdot, t) >$*. When* $\mathcal{H}$ *possess a reproducing kernel, it is said to be a RKHS*

Now we will give some useful propositions which will be used during the work.

Proposition 2.1.1. *If k is a reproducing kernel, then* $k(x, x') =< k(x, \cdot), k(x', \cdot) >, \forall x, x' \in S$*. Now in terms of the canonical feature map if the kernel is defined through the feature map, then* $\phi(x) = k(x, \cdot)$*, therefore we can define the Hilbert spaces through these kernels.*

Finally in [5] Chapter 2 from section 7, and Chapter 6, contains more in depth information about RKHS. We will take from a summary of section 2.7 the following brief:

Being a reproducing kernel is equivalent to being a positive definite function, therefore every positive definite function is the kernel of a unique RKHS. With this set we will dive into MMD which will serve us as an introduction to HSIC which will be the first dependence measure.

2.2. Maximum Mean Discrepancy (MMD)

Previous section was an introduction to RKHS's, now in this section we will use them define a homogeneity test in terms of the embeddings of the probability measures. This will allow us to see each probability distribution as an element of a RKHS. As we saw, this spaces always have an associated distance. Therefore, the test will in general terms consist of applying this distance to the respective mean embeddings. If this distance equals zero, then these probability distributions are homogeneous. Most content for this chapter is taken from [6] and [7].

Now we will introduce what homogeneity is in terms of functions this will be used in order to assure that MMD is an homogeneity test, then we will give a definition for MMD which this section will revolve around.

Lemma 2.2.1 (Homogeneity)**.** *Given two Borel probability measures* $\mathbb{P}$ *and* $\mathbb{Q}$ *are equal if and only if* $\mathbb{E}f(X) = \mathbb{E}f(Y)\ \forall f \in \mathbb{C}(\mathbb{X})(X)\ X \sim \mathbb{P}$ *and* $Y \sim \mathbb{Q}$

Defining a metric which satisfies this is a complex task, in this section our goal is to present one approach which can be taken to define an homogeneity test using mean embeddings in functional spaces.

Let $\mathcal{F}$ be a class of functions f: $X \to \mathbb{R}$ the MMD based on $\mathcal{F}$ is

$$\gamma(\mathbb{P}, \mathbb{Q}) = MMD(\mathcal{F}, \mathbb{P}, \mathbb{Q}) = \sup_{f \in \mathcal{F}} \{\mathbb{E}f(X) - \mathbb{E}f(Y)\}$$

This $\mathcal{F}$ must be rich enough for it to ensure that $\mathbb{P} = \mathbb{Q} \leftrightarrow \gamma(\mathbb{P}, \mathbb{Q}) = 0$. And restrictive enough for the empirical estimate to converge quickly as the sample size increases. As we can see calculating the supremum is not an approachable task, therefore now we will focus on how to calculate this without measuring each function in $\mathcal{F}$.

Now we will introduce the concept of Mean Embedding, which as we anticipated, will help us to define MMD as a distance between two elements of a RKHS. Some concepts which are useful for a complete understanding of this concepts are introduced in Appendix A Section A.1

Definition 2.2.1. ***Mean embedding***

Given a probability distribution $\mathbb{P}$ we will define the mean embedding of $\mathbb{P}$ as an element $\mu_p \in \mathcal{H}$ such that

$$\mathbb{E}(f(X)) =< f, \mu_{\mathbb{P}} >_{\mathcal{H}}, \forall f \in \mathcal{H}$$

Which is equivalent to:

$$\mu_{\mathbb{P}} = \mathbb{E}(k(\cdot, X))$$

This is shown in Appendix A Section A.1

2.2.1. Expressing MMD by mean embeddings

Now that we have defined what a mean embedding is, we will express MMD in terms of mean embeddings, this will help us to manage MMD as a tangible concept and not an abstract entity.

Given the conditions of Lemma A.1.1 ($\mu_{\mathbb{P}}$ and $\mu_{\mathbb{Q}}$ exist) then:

$X \sim \mathbb{P}\mu_{\mathbb{P}} \equiv \mathbb{E}_{X \sim \mathbb{P}}(k(\cdot, X))$ $Y \sim \mathbb{Q}\mu_{\mathbb{Q}} \equiv \mathbb{E}_{Y \sim \mathbb{Q}}(k(\cdot, Y))$

and:

$$MMD(\mathcal{F}, \mathbb{P}, \mathbb{Q}) = \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}$$

This is the first step, now that we've expressed MMD in terms of mean embeddings we will use Definition 2.2.1 to express MMD in terms of expectations of kernels which is something we can compute

Proposition 2.2.2. *Given:* $X, X' \sim \mathbb{P}$ *and* $Y, Y' \sim \mathbb{Q}$ *and X and Y are independent then:*

$$MMD^2(\mathcal{F}, \mathbb{P}, \mathbb{Q}) = \mathbb{E}(k(X, X')) + \mathbb{E}(k(Y, Y')) - 2\mathbb{E}k(X, Y).$$

This is proven in Appendix A Section A.1, in addition in Subsection A.1.1 we've shown that MMD defines an homogeneity test, this is to showcase that this measure is defines an homogeneity test.

2.2.2. Application to independence test

Now that we've introduced MMD we will develop an independence criterion which will be conduced by the following idea: Given $\mathcal{X} \sim \mathbb{P}$ and $\mathcal{Y} \sim \mathbb{Q}$ whose joint distribution is $\mathbb{P}_{\mathcal{XY}}$ then the test of independence between these variables will be determining if $\mathbb{P}_{\mathcal{XY}}$ is equal to the product of the marginals $\mathbb{PQ}$. Therefore:

$\mathcal{MMD}(\mathcal{F}, \mathbb{P}_{\mathcal{XY}}, \mathbb{PQ}) = 0$ if and only if $\mathcal{X}$ and $\mathcal{Y}$ are independent. To characterize this independence test we need to introduce a new RKHS, which is a tensor product of the RKHS's in which the marginal distributions of the random variables are embedded. Let $\mathcal{X}$ and $\mathcal{Y}$ be two topological spaces and let k and l be kernels on these spaces, with respective RKHS $\mathcal{H}$ and $\mathcal{G}$. Let us denote as $\upsilon((x, y), (x', y'))$ a kernel on the product space $\mathcal{X} \times \mathcal{Y}$ with RKHS $\mathcal{H}_\upsilon$. This space is known as the tensor product space $\mathcal{H} \times \mathcal{G}$. Tensor product spaces are explained in detail in Appendix A Section A.1 Subsection A.1.2.

Now we will introduce some definitions which will be used in the next section to define the independence test HSIC and show that it's equivalent to MMD.

Useful definitions for the following content

$$\mathbb{E}_{\mathcal{X}} f(\mathcal{X}) = \int f(x) d\mathbb{P}(x)$$
$$\mathbb{E}_{\mathcal{Y}} f(\mathcal{Y}) = \int f(y) d\mathbb{Q}(y)$$
$$\mathbb{E}_{\mathcal{XY}} f(\mathcal{XY}) = \int f(x, y) d\mathbb{P}_{\mathcal{XY}}(x, y)$$

Using this notation, the mean embedding of $\mathbb{P}_{\mathcal{XY}}$ and $\mathbb{PQ}$ are:

$$\mu_{\mathbb{P}_{\mathcal{XY}}} = \mathbb{E}_{\mathcal{XY}} \upsilon((\mathcal{X}, \mathcal{Y}),)$$
$$\mu_{\mathbb{PQ}} = \mathbb{E}_{\mathcal{XY}} \upsilon((\mathcal{X}, \mathcal{Y}),)$$

In terms of these embeddings:

$$\mathcal{MMD}(\mathcal{F}, \mathbb{P}_{\mathcal{XY}}, \mathbb{PQ}) = \|\mu_{\mathbb{P}_{\mathcal{XY}}} - \mu_{\mathbb{PQ}}\|_{\mathbb{H}_v} \tag{2.1}$$

This last definition will be used in the next section because it'll make really easy the equivalence between HSIC and MMD

2.3. Hilbert-Schmidt Independence Criterion (HSIC)

In the previous section we introduced MMD which was an homogeneity test, now we will use MMD to define an independence test. Originally HSIC was defined as the squared HS norm of the associated cross-covariance operator, but due to this definition being pretty abstract we will manage another definition which will be shown to be equivalent. Appendix A Section A.2 Subsection A.2.1 gives more information about the original definition.

After we will determine whether the dependence returned via HSIC is statistically significant by studying an hypothesis test with HSIC as its statistic and testing it empirically. Finally we will prove the equivalence of the HSIC and MMD applied to $\mathbb{P}_{\mathcal{XY}}$ and $\mathbb{PQ}$. Most information for this section is taken from [8], [9], [10], [11] and [12]

We will start defining HSIC in terms of expectations of kernels and head step by step showing that HSIC expressed this way generates an independence test

If we denote $X, X' \sim \mathbb{P}$ and $Y, Y' \sim \mathbb{Q}$ then:

$$HSIC(\mathbb{P}_{\mathcal{XY}}, \mathcal{H}, \mathcal{G}) = \mathbb{E}_{xx'yy'}[k(x,x')l(y,y')] + \mathbb{E}_{xx'}[k(x,x')]\mathbb{E}_{yy'}[l(y,y')] - 2\mathbb{E}_{xy}[\mathbb{E}_{x'}[k(x,x')]\mathbb{E}_{y'}[l(y,y')]] \tag{2.2}$$

In the Appendix A Section A.2 is shown the proof that this definition is equivalent to the squared HS norm of the associated cross-covariance operator

In order for HSIC to generate an independence test we will need that given two random variables $X \sim \mathbb{P}$ and $X \sim \mathbb{Q}$, with joint distribution $\mathbb{P}_{XY}$, and two RKHS's $\mathcal{H}$ and $\mathcal{G}$ with characteristic kernels k and l, then HSIC($\mathbb{P}_{XY}, \mathcal{H}, \mathcal{G}$) = 0 if and only if $\mathbb{P}_{XY} = \mathbb{PQ}$, which is equivalent of X and Y being independent.

We won't prove this proposition because we've already proven in Appendix A Section A.1 Subsection A.1.1, that MMD defines an homogeneity test, and we will use that HSIC and MMD are equivalent, therefore if for MMD it is true then for HSIC is true as well.

Now we will take a look at this new definition for HSIC and the definition of MMD given in equation 2.1 it's easy to proove that MMD and HSIC are equivalent. (To prove it one just needs to express the

kernel υ in terms of the respectives kernels k and l of $\mathcal{H}$ and $\mathcal{G}$ and unravel the norm in terms of the expectations).

2.3.1. HSIC associated statistic

In the previous subsection we talked about the theoretical expression for HSIC, but as our objective is to implement this test and work with it, this subsection will show how can we calculate this statistic numerically. Our objective for this subsection is to prove that the next expression is in fact the empirical HSIC.

Empirical HSIC

$$HSIC(\mathbb{P}_{\mathcal{X}\mathcal{Y}}, \mathcal{H}, \mathcal{G}) = (m-1)^{-2}\mathbf{tr}KHLH \tag{2.3}$$

where: $H, K, L \in \mathbb{R}^{m\times m}$, $K_{i,j} = k(x_i, y_j)$, $L_{i,j} = l(x_i, y_j)$ and $H_{i,j} = \delta_{i,j} - m^{-1}$

In the Appendix A Section A.2 Subsection A.2.3 is shown the proof of this theorem.

Now we will briefly present the real asymptotic distribution of HSIC

Theorem 2.3.1. *Under the $\mathcal{H}_0$ the U-statistic HSIC corresponding to the V-statistic*

$$HSIC(Z) = \frac{1}{m^4}\sum_{i,j,q,r\in_4^m} h_{ijqr}$$

is degenerate, meaning $\mathbb{E}h = 0$. In this case, HSIC(Z) converges in distribution according to [12], section 5.5.2

$$mHSIC(Z) \to \sum_{l=1} \lambda_l z_l^2$$

where $z_l \sim \mathcal{N}(0,1)$ i.i.d and λ_l are the solutions to the eigenvalue problem

$$\lambda_l \psi_l(z_j) = \int h_{ijqr}\psi_l(z_i)dF_{iqr}$$

where the integral is over the distribution of variables z_i, z_q and z_r [8]

As it's easy to see, this expression isn't manageable because we will need to compute the distribution for every distribution we were to test, and if we were to test samples from some distribution then all we could compute would be an estimate of this asymptotic distribution, therefore now we will introduce an easy way of calculating the $1-\alpha$ quantile of the null distribution which is what we need for hypothesis

testing.

Approximating the $1-\alpha$ quantile of the null distribution

A hypothesis test using HSIC(Z) could be derived from Theorem 3.3 above by computing the $(1-\alpha)$th quantile of the distribution $\sum_{l=1} \lambda_l z_l^2$, where consistency of the test (that is, the convergence to zero of the Type II error for $m \to \infty$) is guaranteed by the decay as m^{-1} of the variance of HSIC(Z) under H_1 . The distribution under H_0 is complex, however: the question then becomes how to accurately approximate its quantiles.

One approach taken by [8] is by using a Gamma distribution, which as we can see in the figure underneath is quite accurate.

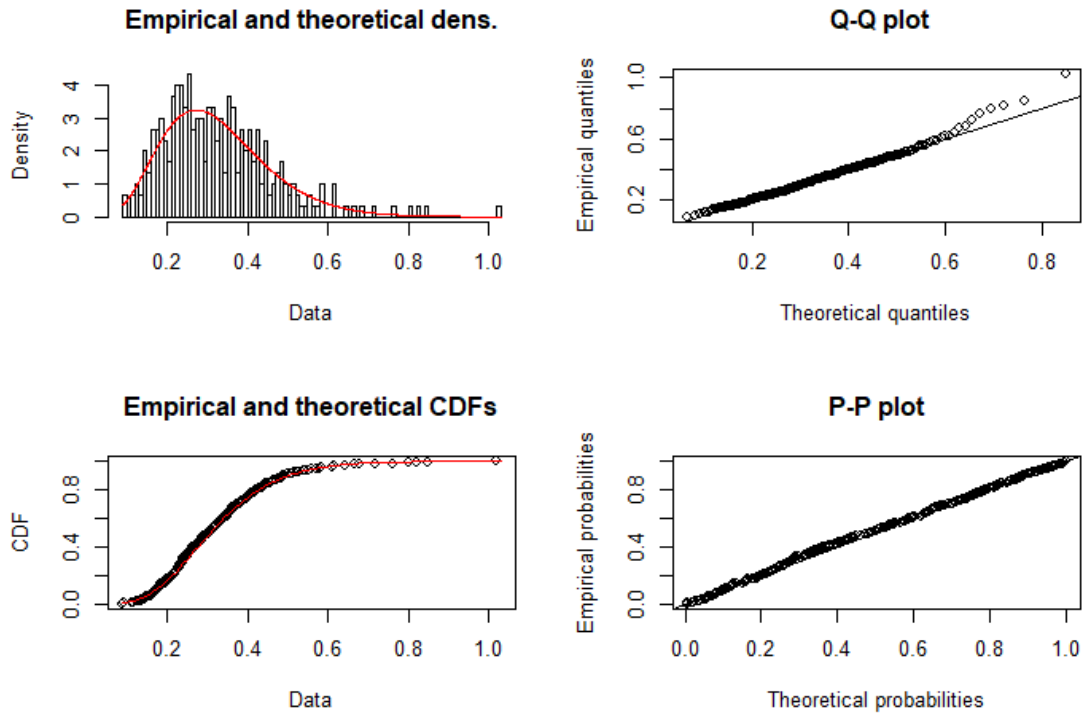


Figure 2.2: Asymptotic comparison of the HSIC statistic and the associated Gamma distribution

2.4. Energy Distance

As we did for the previous sections we will an homogeneity test based on energy distance, then we will use it to formulate another independence test, distance covariance and distance correlation. This measure revolves from the idea of measuring the distance of CDF's, in one dimension this can be pretty easily calculated, but as the dimensions grow the complexity grows with it. Therefore, we will

use characteristic functions, which will allow us to define a $\mathcal{L}_2$ distance in terms of the characteristic functions, which comes with the benefits of always existing and being fairly easy to manage in any dimension.

This test is one of the most popular nowadays because of its power and the fact that it does not depend on any parameter. Most of the content of this section is taken from [13] and [14]. We will start giving a brief introduction for this statistic, then head to it's theoretical formula, give it's associated statistics and conclude with it's convergence in order to ensure that are well defined.

2.4.1. Definitions

In this subsection we will introduce some Definitions which will be used to build the independence test, this test is a generalization of the characteristic function of each distribution.

The following proposition we will show how to measure the distance between two CDF's, this will be used to create the concept of energy distance.

Proposition 2.4.1. *Let $\mathcal{F}$ and $\mathcal{G}$ be two CDFs of the independent random variables X,Y respectively and X',Y' two iid copies of them, then:*

$$2\int_{-\infty}^{\infty}(\mathcal{F}(x)-\mathcal{G}(x))^2dx = 2\mathbb{E}|X-Y| - \mathbb{E}|X-X'| - \mathbb{E}|Y-Y'|$$

In the Appendix A Section A.3 is shown the proof of this proposition.

now we will give the equivalent for any two random variables: Let X and Y be random variables in $\mathbb{R}^d$ with $\mathbb{E}\|X\|_d + \mathbb{E}\|Y\|_d < \infty$ the energy distance between X and Y is defined as:

$$\varepsilon(X,Y) = 2\mathbb{E}\|X-Y\|_d - \mathbb{E}\|X-X'\|_d - \mathbb{E}\|Y-Y'\|_d \tag{2.4}$$

where X' and Y' are i.i.d copies of X and Y respectively. This definition has only one problem, in dimensions greater than one we do not have an equivalence between this distance and the differences between CDF's, therefore we will define it in terms of the characteristic functions. In fact, it can be seen as a weighted $\mathcal{L}_2$ distance between characteristic functions.

Proposition 2.4.2. *Given two independent d-dimensional random variables X and Y, with distributions $\mathbb{P}$ and $\mathbb{Q}$ respectively such that $\mathbb{E}\|X\|_d + \mathbb{E}\|Y\|_d < \infty$ he energy distance between X and Y can be written as:*

$$\varepsilon(X,Y) = \frac{1}{c_d}\int_{\mathbb{R}^d}\frac{|\phi_{\mathbb{P}}(t)-\phi_{\mathbb{Q}}(t)|^2}{\|t\|_d^{d+1}}$$

where

$$c_d = \frac{\pi^{\frac{d+1}{2}}}{\Gamma(\frac{d+1}{2})}$$

being $\Gamma(\cdot)$ the gamma function

In the Appendix A Section A.3 is shown the proof of this proposition.

It is easy to see that the energy distance only vanishes when the distributions are equal, since it is equivalent to having equal characteristic functions.

2.4.2. Application to an independence test

In this subsection we will use the knowledge acquired above to develop a new independence test. This new test is called distance covariance (DCOV), it's name comes from the fact that it is a generalization of the classical product-moment covariance.

We will start by defining the independence test. Given the random vectors $X \in \mathbb{R}^{dx}, Y \in \mathbb{R}^{dy}$, distributions $\mathbb{P}_X$ and $\mathbb{P}_Y$ respectively.Let $\phi_{\mathbb{P}_X}, \phi_{\mathbb{P}_Y}$ denote their characteristic functions and $\phi_{\mathbb{P}_{XY}}$ the characteristic function of the joint distribution. X and Y are independent if and only if $\phi_{\mathbb{P}_X}\phi_{\mathbb{P}_Y} = \phi_{\mathbb{P}_{XY}}$. The covariance energy test is based on measuring a distance between these functions.

First we need to generalize the energy distance expression for random vectors of different dimensions. As defined earlier this expression is obtained from a weighted $\mathcal{L}_2$-distance, imposing rotation invariance and scale equivariance, the energy distance is:

$$\varepsilon(X,Y) = \frac{1}{c_{d_x}c_{d_y}} \int_{\mathbb{R}^{d_x+d_y}} \frac{|\phi_{\mathbb{P}}(t) - \phi_{\mathbb{Q}}(t)|^2}{\|t\|_{d_x}^{d_x+1}\|s\|_{d_y}^{d_y+1}} dtds$$

Where c_d is defined as before. The distance covariance is defined by replacing $\phi_{\mathbb{P}}$ and $\phi_{\mathbb{Q}}$ in the previous formula with characteristic functions of the joint distribution and the product of the marginals respectively. Like the classical product-moment covariance it has associated a correlation, we will now define our two statistics:

Definition 2.4.1. *The distance covariance, DCOV, between random vectors X and Y, with* $\mathbb{E}\|X\|_{d_x} + \mathbb{E}\|Y\|_{d_y} < \infty$*, is the nonnegative number* $\nu^2(X,Y)$ *defined by:*

$$\nu^2(X,Y) = \|\phi_{\mathbb{P}_{X,Y}}(t,s) - \phi_{\mathbb{P}_X}(t)\phi_{\mathbb{P}_Y}(s)\|_w^2 = \frac{1}{c_{d_x}c_{d_y}} \int_{\mathbb{R}^{d_x+d_y}} \frac{|\phi_{\mathbb{P}_{X,Y}}(t,s) - \phi_{\mathbb{P}_X}(t)\phi_{\mathbb{P}_Y}(s)|^2}{\|t\|_{d_x}^{d_x+1}\|s\|_{d_y}^{d_y+1}} dtds$$

The distance correlation, DCOR, between random vectors X and Y, with $\mathbb{E}\|X\|_{d_x} + \mathbb{E}\|Y\|_{d_y} < \infty$*, is the nonnegative number* $\mathcal{R}(X,Y)$ *defined by:*

$$\mathcal{R}(X,Y) = \begin{cases} \frac{\nu^2(X,Y)}{\sqrt{\nu^2(X)\nu^2(Y)}} \text{ if } \nu^2(X)\nu^2(Y) > 0 \\ \\ 0 \text{ if } \nu^2(X)\nu^2(Y) = 0 \end{cases}$$

The distance covariance, like the energy distance, can be expressed using expectations, which will

leave to a way more manageable statistic for it's numerical computation:

Let (X,Y),(X',Y'),(X",Y") $\sim \mathbb{P}_{XY}$ be iid copies of (X,Y), it holds that:

$$\begin{aligned}\nu^2(X,Y) = \mathbb{E}_{XY}\mathbb{E}_{X'Y'}\|X-X'\|_{d_x}\|Y-Y'\|_{d_y} + \mathbb{E}_X\mathbb{E}_{X'}\|X-X'\|_{d_x}\mathbb{E}_Y\mathbb{E}_{Y'}\|Y-Y'\|_{d_y} \\ - 2\mathbb{E}_{XY}[\mathbb{E}_{X'}\|X-X'\|_{d_x}\mathbb{E}_{Y'}\|Y-Y'\|_{d_y}]\end{aligned} \tag{2.5}$$

This proof is similar to the one of A.3, therefore we will leave it for the interested readers.

2.4.3. Energy distance, DCOV and DCOR associated statistics

Now that the theory has been established we will give some estimators for both energy distance and distance covariance. Since we are interested in testing independence we will focus on the DCOV estimator. We will start with an estimator of energy distance, which as it's explained above it's a homogeneity test. Given the definition of energy distance 2.4 now we will define it's statistic as: Given two independent random samples $x = (x_1, ..., x_n)$ and $y = (y_1, ..., y_m)$, the two sample energy statistic corresponding to $\varepsilon(X, Y)$ is:

$$\varepsilon_{n,m}(x,y) = \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\|x_i - y_j\| - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\|x_i - x_j\| - \frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\|y_i - y_j\|$$

Finally, an estimator of the distance covariance can be obtained directly from 2.5.For a random sample $(x, y) = ((x_1, y_1), ..., (x_n, y_n))$ of iid random vectors generated from the joint distribution of $X \in \mathbb{R}^{dx}$ and $Y \in \mathbb{R}^{dy}$, we obtain:

$$\begin{aligned}\nu^2(X,Y) = \frac{1}{n^2}\sum_{i,j=1}^{n}\|x_i - x_j\|_{d_x}\|y_i - y_j\|_{d_y} + \frac{1}{n^2}\sum_{i,j=1}^{n}\|x_i - x_j\|_{d_x}\frac{1}{n^2}\sum_{i,j=1}^{n}\|y_i - y_j\|_{d_y} \\ - \frac{2}{n^3}\sum_{i=1}^{n}[\sum_{j=1}^{n}\|x_i - x_j\|_{d_x}\sum_{j=1}^{n}\|y_i - y_j\|_{d_y}]\end{aligned} \tag{2.6}$$

As we can see this estimate cost is $O(n^2)$, that's the reason we won't calculate the distance covariance this way, our new approach will go as follows:

First we compute the Euclidean distance matrix of each sample, computing all the pairwise distances between sample observations:

$$(a_{ij}) = (\|x_i - x_j\|_{dx}), (b_{ij}) = (\|y_i - y_j\|_{dy}).$$

an easy way to compute this matrix is:

$$A_{ij} = a_{ij} + \overline{a_{i\cdot}} - \overline{a_{\cdot j}} + \overline{a}, for i, j = 1, ..., n$$

where:

$$\overline{a_{i\cdot}} = \frac{1}{n}\sum_{k=1}^{n} a_{ik}, \overline{a_{\cdot j}} = \frac{1}{n}\sum_{k=1}^{n} a_{kj}, \overline{a} = \frac{1}{n^2}\sum_{k=1}^{n}\sum_{l=1}^{n} a_{lk}$$

equivalently for B. In terms of these matrix, the distance covariance $\nu^2(x, y)$ is:

$$\nu^2(x, y) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}B_{ij}$$

Finally the distance correlation is:

$$\mathcal{R}(X, Y) = \begin{cases} \frac{\nu_n^2(x,y)}{\sqrt{\nu_n^2(x)\nu_n^2(y)}} \text{ if } \nu_n^2(x)\nu_n^2(y) > 0 \\ \\ 0 \text{ if } \nu_n^2(x)\nu_n^2(y) = 0 \end{cases}$$

where:

$$\nu_n^2(x) = \nu_n^2(x, x) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}$$
$$\nu_n^2(y) = \nu_n^2(y, y) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} B_{ij}$$

Now all thats left to see is that this statistics converge almost surely when the random vectors have finite first moments which is: if $\mathbb{E}\|X\| + \mathbb{E}\|Y\| < \infty$ then

$$\lim_{n\to\infty} \nu_n^2(x, y) \xrightarrow{a.s} \nu^2(X, Y)$$

In Appendix A Section A.3 Subsection A.3.1 is proven the convergence.

Asymptotic properties of nν^2

As in the previous section for HSIC, if we want to create an independence test, we will need to study it's asymptotic distribution, which will help us create an efficient independent test.

In [13] Section 2.4 *Asymptotic properties of* $n\nu^2$, it is proven that if $\mathbb{E}\|X + Y\| < \infty$ then $\nu^2\frac{n}{S2}$ converges in distribution to a quadratic form.

$$Q =^D \sum_{j=1}^{\infty} \lambda_j Z_j^2 \tag{2.7}$$

where Z_j are independent standard normal random variables, aλ_j are non-negative constants that depend on the distribution of (X,Y), and $E(Q) = 1$.

As calculating this distribution is not efficient, in order to create a asymptotic test with a significance

level we will use that: the tail of the quadratic form it's similar enough to the one of a Chi squared with one degree of freedom.

$$\nu^2 \frac{n}{S2} \geq \mathcal{X}^2_{1;1-\alpha}$$

where $\mathcal{X}^2_1$ denotes a Chisquared distribution with one degree of freedom, and let $0 < \alpha \leq 0{,}215$. This result is taken from [15], page 181.

2.5. Randomized Dependence Coefficient (RDC)

In this section we will give our final aproach to measuring dependencies, taken from [16] and [17]. As always we will start with a brief introduction about this dependence measure, then as this measure is an approximation as we will see, we will head to it's statistic and how to calculate it. This statistic is a scalable and easy to implement estimator of the following:

Definition 2.5.1. *The Hirschfeld-Gebelein-Rényi's Maximum Correlation Coefficient (HGR) was defined by Gebelein in 1941 [3] as:*

$$hgr(X, Y) = \sup_{f,g} \rho(f(X), g(Y)) \tag{2.8}$$

Where f and g are any Borel-measurable function with finite variance. Given that this estatistic the search of a supremum in an infinite-dimensional space in practice may be really difficult to compute.

The Randomized Dependence Coefficient (RDC) measures the dependence between the random variables $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}^q$ as the largest canonical correlation between the k random non-linear projections of the copula transformation of the variables

Now we will give the introduction of two concepts which will allow us to understand the measure, being: what is a copula, how to calculate the copula, is it consistent and finally what is canonical correlation. With this few new concepts we will give a brief definition of the measure.

Definition 2.5.2 (Copula)**.** *Given a d-dimensional random vector$X_1, ..., X_d$ be iid random variables with cumulative distribution functions $F_1, ..., F_d$, the vector U = ($U_1, ..., U_d$) = ($F_1(X_1)$,...,$F_d(X_d)$) whose marginals are U[0,1], is known as the copula transformation*

The proof for this definition is found in Appendix A Section A.4.

In practice, the estimation of univariate cdfs is easily done given a few hundred of observations. In addition, cdfs converge uniformly to the true distribution, which is shown with this next theorem.

Theorem 2.5.1. *Glivenko-Cantelli*

Let $X_1, ..., X_n$ be iid random variables with common cumulative distribution function P. Then, the empirical cumulative distribution function, defined as

$$P_n := \frac{1}{n} \sum_{i=1}^{n} I - (X_i \leq x)$$

converges uniformly to P:

$$\|P_n - P\|_\infty = \sup_{x \in R} |P_n(x) - P(x)| \rightarrow^{a.s} 0$$

The proof for this definition is found in Appendix A Section A.4.

As we've seen, in order to calculate the RDC statistic, we need to compute the copula transformation using the empirical cumulative distribution function, which we've seen it's pretty consistent and simple. Then we need to augment these empirical transformations with non-linear projections, so that linear methods can be used to measure non-linear dependences in the original data. In [16] sin and cosine projections were chosen:

$$\Phi(\mathbf{X}; k, s) := \begin{pmatrix} \phi(w_1^T x_1 + b_1) & ... & \phi(w_k^T x_1 + b_k) \\ . & . & . \\ . & . & . \\ . & . & . \\ \phi(w_1^T x_n + b_1) & ... & \phi(w_k^T x_n + b_k) \end{pmatrix}$$

Where $\phi(x) = (cos(x), sin(x))$, $W \in \mathbb{R}^k$, $W = w_1, ..w_k$ iid and $W \sim \mathcal{N}(0, s)$ and B = $b_1, ..., b_k \sim U[-\pi, \pi]$. Choosing W to be Gaussian is equivalent to using a Gaussian kernel for the projections, and the parameter s is equivalent to the kernel width.

2.5.1. Canonical Correlation Analysis(CCA)

As we've seen in the definition 1.4.1 we need to maximize the correlation between the variables for any pair of given functions which of $\mathcal{L}_2$, here is where we will use CCA, which maximize the correlations for Φ for any given real valued vectors. For this subsection, most content is taken from [18] Chapter 14.

Let us consider the correlation $\rho(a, b)$ between the two projections in more detail. Suppose that:

$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim \left(\begin{pmatrix} \mu \\ \nu \end{pmatrix} \quad \begin{pmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{pmatrix} \right)$$

Then:

$$\rho(a,b) = \frac{a^T \Sigma_{XY} b}{(a^T \Sigma_{XX} a)^{1/2} (b^T \Sigma_{YY} b)^{1/2}}$$

Which is easy to see that: $\rho(a,b) = \rho(ca,b)$ for any c. Given the invariance of scale, we may rescale projections a and b, leaving with the equivalent problem of :

$max_{a,b}(a^T \Sigma_{XX} b)$ under the constrains: $a^T \Sigma_{XX} a = 1$ and $b^T \Sigma_{YY} b = 1$.

First we need to define: $\mathcal{K} = \Sigma_{XX}^{-1/2} \Sigma_{XY} \Sigma_{YY}^{-1/2}$ which its eigenvalues will be the canonical correlation coefficients. This eigenvalues can be easily obtained through the singular value decomposition of $\mathcal{K}$. This canonical correlation coefficients are the correlations between the random projections $\alpha^T \Phi(\mathbb{P}(X); k, s)$ and $\beta^T \Phi(\mathbb{Q}(Y); k, s)$. Therefore the maximum of this coefficient will be the supremum which we were searching.

To sum it all up we will give a final and cohesive definition for RDC:

Definition 2.5.3. *Given two d-dimensional random variables $X \sim \mathbb{P}$ and $Y \sim \mathbb{Q}$, and parameters $k \in \mathbb{N}$ and $s \in \mathbb{R}$ (n,s >0), the Randomized Dependence Coefficient between the variables is defined as:*

$$rdc(X, Y; k, s) = \sup_{\alpha,\beta} \rho(\alpha^T \Phi(\mathbb{P}(X); k, s), \beta^T \Phi(\mathbb{Q}(Y); k, s)$$

To conclude, we will introduce it's asymptotic distribution, taken from [16] Section 4 Subsection **Testing for independence with RDC**.

$$(\frac{2k+3}{2} - n) log \prod_{i=1}^{k} (1 - \rho_i^2) \sim \mathcal{X}_{k^2}^2 \tag{2.9}$$

where $\rho_1, ..., \rho_k$ are the canonical correlations between the two sets of non-linear projections. Figure below shows how accurate this asymptotic behavior is

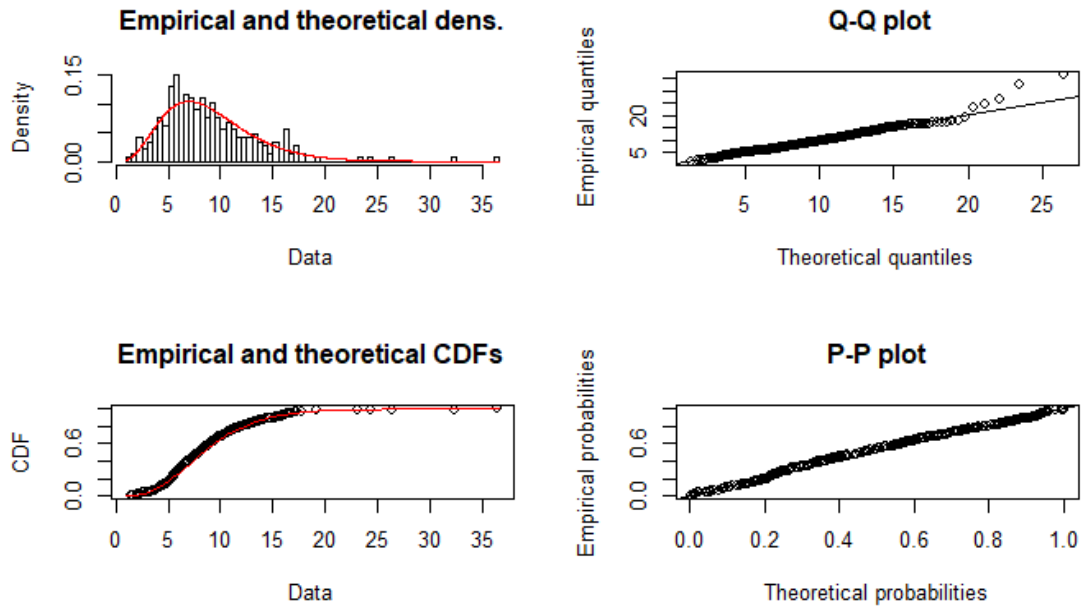


Figure 2.3: RDC statistic with a chi-squared distribution with 9 degrees of freedom

3

Design

In the previous chapter we've explained the tests that we will be developing. In this section we will explain the software design process, starting with analysis of the tests in order to provide a general overview of the problem for a better understanding of the choices taken. As we've explained our goal for this project is to study three different independence tests, implement them and compare them in order to determine in which scenarios a test should be preferred over the others. With this in mind it's easy to see that this experiment in the future may be used again with more independence test as they shall arise in the scientific field.

3.0.1. Requirement analysis

As we said in the previous introduction to the chapter the biggest requirement in our project is for it to be scalable and easy to modify, the scope of being scalable includes not only the structure of the testing, where adding new tests needs to be as easy as possible, but it goes as far as the different types of datasets that may be used, shall they be a function which generates them or a database; the experiments which will be performed to the tests, should they study how any parameter may affect the performance of the independence tests, for example: studying how varying the sample size of the data may affect the overall performance, applying different functions to the data, such as rotating the data, and seeing how adding small variations to the data like Gaussian noise may affect the measures.

On the grounds that we need to measure performance differences between methods all software must be developed in the same language assuring that the obtained results come from the actual algorithm and not the language difference, for example if one algorithm were to be implemented in C and another in Python and we were to analyze time performance of our algorithms, the results wont be conclusive because the differences found may be produced only because a good programmer in C will manage the memory and the data access way more accurately than the one in python.

Other aspect to take into account is that given the volume of data that we will be working with efficiency is of key importance, therefore parallelization will be heavily used to ensure a fast process of data. This requirement implies much more than only being efficient, as mentioned before, our software

needs to be scalable, as we may add new tests, measures, and data, we will need to make the software as modular as possible in order to avoid complications in the future and handle the possible growth of the project.

3.0.2. Project architecture

With all the requirements specified in the previous section, now we will explain the design chosen to implement our project.

All our software is built around two classes: IndependenceTest and IndependenceTestTester. Both being abstract classes which held the code for the independence tests and the experiments respectively 3.1 shows the class diagram of our project.

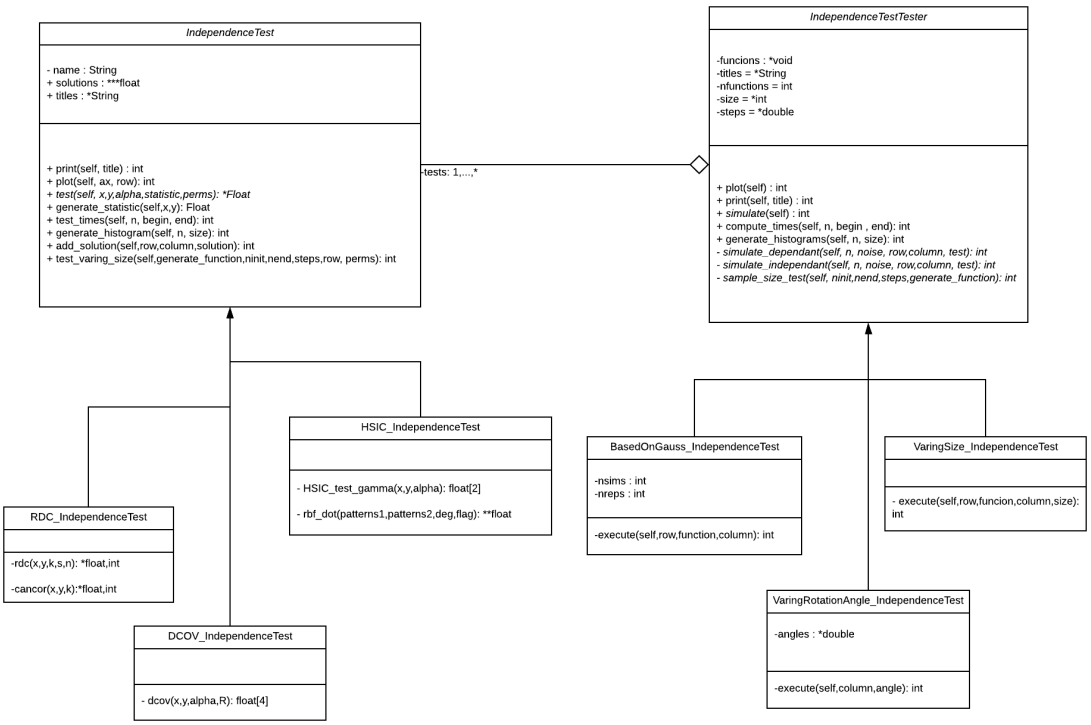


Figure 3.1: Class diagram

IndependenceTest

This abstract class helds the main core which all independence tests will inherit.

As one of the requirement is for the software to be modular, all tests will control their own data and the progress of the experiments within themselves, the internal variables are the following: Name,a string containing the name of the test, this will be automatically given to the mother class constructor by the child's constructors, this will help the implementation of plotting, which will also be held by the

IndependenceTests. Solutions, a matrix of floats which will include the progress of the experiment. The child's constructor will need to know the number of different variations which will be performed and the number of different data sets which will be used, because they will be the number of columns and rows of this matrix respectively. Titles, as mentioned before, this class will held the implementation of the plots within itself, this variable contains the title of each subplot.

This design will allow us an easy parallelization,as will be shown in capítulo 4, where we will dive in depth on how we parallelized our project.

All functionality will be held in this abstract class containing all functionality, such as plotting the results of the experiment for a given test, computing the time cost of an experiment and generating an empirical histogram of the statistic. The specific implementation of the test will be held in an abstract function called test which will be implemented in each child object.

To sum things up for this class,Figure 3.2 will be included the sequence diagram for plotting, which showcase the general idea of how any functionality will be implemented besides the experiments, which are more complex, Figure 3.3 contains the sequence diagram of any experiment which is the only process which differs from the general one shown in 3.2.

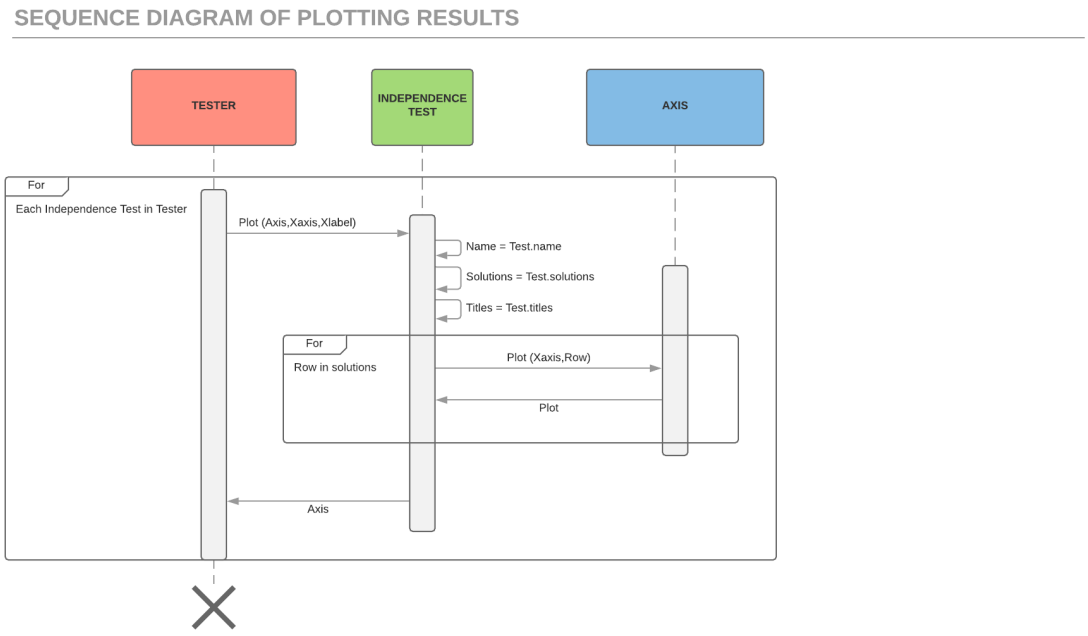


Figure 3.2: Sequence diagram for plots

IndependenceTestTester

As we've seen, while IndependenceTest helds the implementation for the dependence measures, this abstract class implements the general functionalities of all the performed experiments. As we've said, this software needs to be scalable for any amount of dependence measures which may not be

implemented now, that's why we decided to create that abstract class, while each test may differ as much as needed one from another, all are forced to follow the same skeleton, allowing us to provide to our tests a list of IndependenceTest which are transparent to the class IndependenceTestTester, and the experiment will be performed by calling the function test of each instance, which will be overwritten by the child instance with each desired dependence measure.

This class receives: a list of IndependenceTests, a list of functions which will be used to generate data, this functions may be calls to a data base or functions which generate the data through random samples of a mixture of known probability distributions, the only requirement of this functions is for them to only have one parameter which is the sample size.

The variables step and size are stored because each test needs the sample size which will be used, as this parameter may vary it's stored as a pointer to int (array). Step is the number of variations which will be performed.

Finally in order to ensure the minimum amount of repeated code the functionality of measuring the power of a test given X and Y is implemented in a function called simulate, letting the task of modifying the datasets as needed to each child object. Figure 3.3 shows a sequence diagram of an experiment.

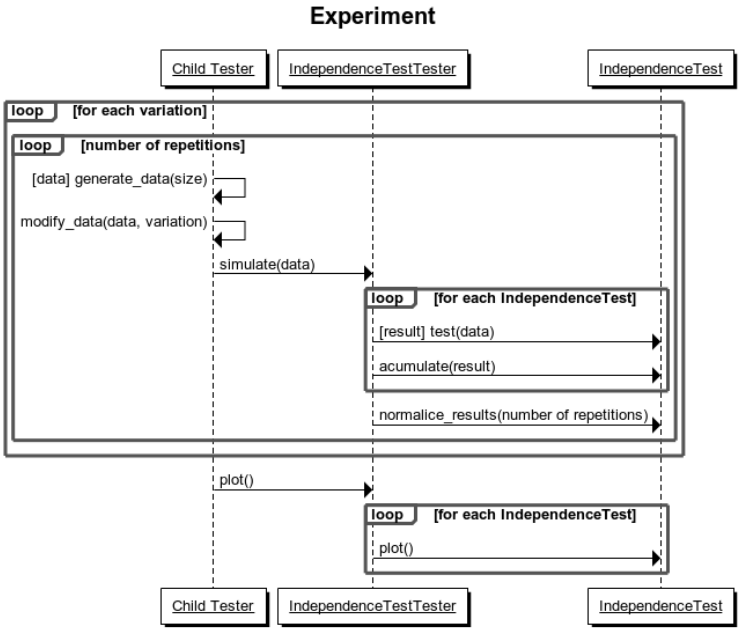


Figure 3.3: Sequence diagram for experiments

In the following section we will dive into the process of how this software was implemented, the main problems we found along the way and how we solved them.

4

Development

In the previous section we presented the design aspects of our project, now we will introduce our development experience and the reasons behind each decision. First of all we will start by how we achieved the main goals of our software, being: efficiency, modularity and scalability. In each of this subsections we will dive into subjects such as how we parallelized, which language was used and which tools were applied. We will follow with specific details about each independence test implementation, including problems which rose during implementation. Finally we will conclude this chapter explaining how we kept track of the progress, backups testing and the project management.

4.1. General aspects of implementation

As this software includes a heavy mathematical component we decided to implement all the functionality in Python, due to the amount of already existing mathematical libraries such as numpy and scipy, which were really helpful during the implementation of all functionality. The graphical display was mostly implemented in Python although all plots shown comparing distributions were plotted in R, this is because of the simplicity which R provides to perform plots comparing distributions. This will be explained in detail in 4.2.2. Now we will dive into how we achieved each general requirement of our project.

4.1.1. Code efficiency

In this subsection we will explain how we achieved efficiency in this project, diving into parallelization and libraries used. The efficiency in our implementation, as memory in this project is generally a problem, is generally related to time efficiency, which is the key factor which might detriment the performance of this measures and make us choose one over other.

As our project was implemented in python, we had access to numpy and scipy, which are libraries implemented in low levels languages, like C, Fortran and Cython, making them really efficient. Therefore, we used them whenever it was possible, this also allowed us to use matrix calculus with numpy arrays, which not only makes the code easier to read, but it also makes it much more efficient reducing

the number of for loops and operations. Code 3.1 shows how we calculate the hyper-parameter for the Gaussian kernel in HSIC as an example of what we mean in the previous sentence.

Code 4.1: Sample of how to calculate the median of the distances for a sample $x \in \mathbb{R}^n$

```
size = len(x)
G = np.sum(x*x,1) #Here we calculate the square of each sample
Q = np.repeat(G,size).reshape(size,size) #row i contain each the square of sample i n times
R = Q.T #colum i contain each the square of sample i n times
dists = Q + R -2*np.dot(xmed,xmed.T) #we calculate (x -y)^2 = x^2+y^2-2x*y
dists = dists -np.tril(dists) #we remove repeated distances (x-y)^2 = (y-x)^2
dists = dists.reshape(size*size,1)
hyperparameter = np.sqrt(0.5*np.median(dists)) #Calculate the hyperparameter of our kernel
```

In addition to the code, parallelization was key in our project in order to obtain results in a reasonable time span. All parallelization was created by threads, given the amount of parallel lines of code we were managing, and how little work each had to made, creating process for each line wouldn't have been optimal due to the time it takes to create a new process. For most parallel process we generated a thread pool with the library concurrent.futures [1] which provides a high level interface for asynchronously executing callables, which makes most of our simulations something as simple as shown in Code 3.1, which presents how to create an histogram of the statistic of all independence tests included in our independence test tester and the amount of cores we will be using. The parameter of max workers is fixed to the maximum between the amount of independence tests we have and the number of cores of the computer. This is because if we are working with less threads than the number of cores there's no need to allow for more workers to be active, while if there are more independence tests than cores, then the computation time will decrease to the number of cores and increase afterwards, this is because of the cost of context switching. In Appendix B Section B.2 we show a small experiment to showcase this.

Code 4.2: Code sample of how to create a thread pool with concurrent.futures

```
with concurrent.futures.ThreadPoolExecutor(max_workers= min(len(self.tests),ncores)) as executor:
            futures = {executor.submit(test.generate_histogram,sample_size) for test in self.tests}
            concurrent.futures.wait(futures)
```

4.1.2. Modularity and scalability

Another requirement was for our software to be as modular and scalable as possible, this comes due to the fact that this experiments may be used again in the future whether other dependence measures may be compared. In order to ensure that our system was as modular as possible, we followed the design presented in Figure 3.1. For the implementation of the abstract classes in python we used the

[1] https://pythonhosted.org/futures/

modules abstractmethod and ABCMeta from the abc package. Allowing for all main code of the test to be storaged in the child class while making all the experiments transparent to the implementation underneath. In Code 4.1.1 is shown how for any independence test the calling maintains the same.

4.2. Specific details about each independence test implementation

We've already discussed the general aspects of the implementation, now we will dive into specific details that rose for each dependence measure, such as problems which rose during the implementation, choices taken in each algorithm

4.2.1. RDC

As explained in section 2.5 the parameter k will improve the performance of the test the largest it is, but due to numerical issues, if k is too large,then rank($\Phi(X)$) <k or rank($\Phi(Y)$) <k, so we need to find the largest k such that the eigenvalues, solutions of the canonical correlation analysis, are real-valued. As this is a problem dependent of the data, we will preform a binary search for the largest k which meets the condition. As the complexity of RDC is $O(k^2 n)$ adding a binary search to the algorithm wont affect its overall complexity as the complexity of the binary search is $O(log(k))$, which is irrelevant.

In order to compute RDC, we needed to calculate the canonical correlation analysis, which is not implemented in python. In the Appendix B Section B.3 we present how we calculate the canonical correlation analysis in python.

4.2.2. HSIC

We've decided to implement HSIC following the Mathlab implementation [2] which makes usage of a Gaussian kernel :

$$K(x, y) = exp(-\frac{\|x - y\|^2}{\mu^2})$$

where μ is the median of the euclidean distances between samples. This kernel will be used because this kernel tends to give smooth functions, which are easier to work with. In depth explanation is presented in Appendix B section B.1

[2] http://www.gatsby.ucl.ac.uk/ gretton/indepTestFiles/indep.htm

4.2.3. Plots

As in this work we have been working intensively with probability distributions, in order to ease the task of testing hypothesis and showcasing the asymptotic behavior of our statistics, we decided to make use of R in our project. All data was collected from the experiments performed in python, in R we performed K-S test to each statistic with different sample sizes and variations which will be explained in detail in chapter 5 and saw how good our null hypothesis was. In the appendix we showcase some results obtained with R, for example Figure C.15 shows how good of a fit is a Gamma distribution to the HSIC distribution.

4.3. Version control, repositories and continuous integration

In order to maintain control of each change throughout the project we needed to use tools in order to manage and control the advance of the project.

As a version control system we used git, which provides simplicity and comes with the advantage that hosting services for version control using git like GitHub exist. We chose GitHub because it is free and comes with all functionality for public repositories, one key functionality is that provides a version history of your code, so that previous versions are not lost with every new merge, easing removing mistakes or going back to a previous version if necessary.

In addition of version control we used Travis-CI [3] in order to automatically test all changes made. Travis-CI [4] is a hosted continuous integration service used to build and test software projects hosted at GitHub, furthermore this tool is free for open projects.

Travis-CI is configured by adding a file named *.travis.yml* to the root of the repository, specifying programming language used, the desired building and testing environment, the script to run the tests, when and what to do whenever a petition is made to the repository. Code 3.4 shows an example of a *.travis.yml*.

To sum up this section Figure 4.1 presents graphically the development process of this project showcasing the different tools used and their relations.

[3] https://travis-ci.org/

[4] https://github.com/travis-ci/travis-ci/blob/master/README.md

Code 4.3: yml file used in order to incorporate Travis-CI in our repository.

```
language: python
python:
  -"2.7"
  -"3.4"
install:
  -pip install unittest2
script:
  python tests/HSIC_test.py
  python tests/RDC_test.py
  python tests/DCOV_test.py
  python tests/IndependenceTest_test.py
  python tests/IndependenceTestTester_test.py
notifications:
  email:
    recipients:
      -roberto.alcover@estudiante.uam.es
      -robertoalcovercouso@gmail.com
    on_success: never # default: change
    on_failure: always # default: always
```

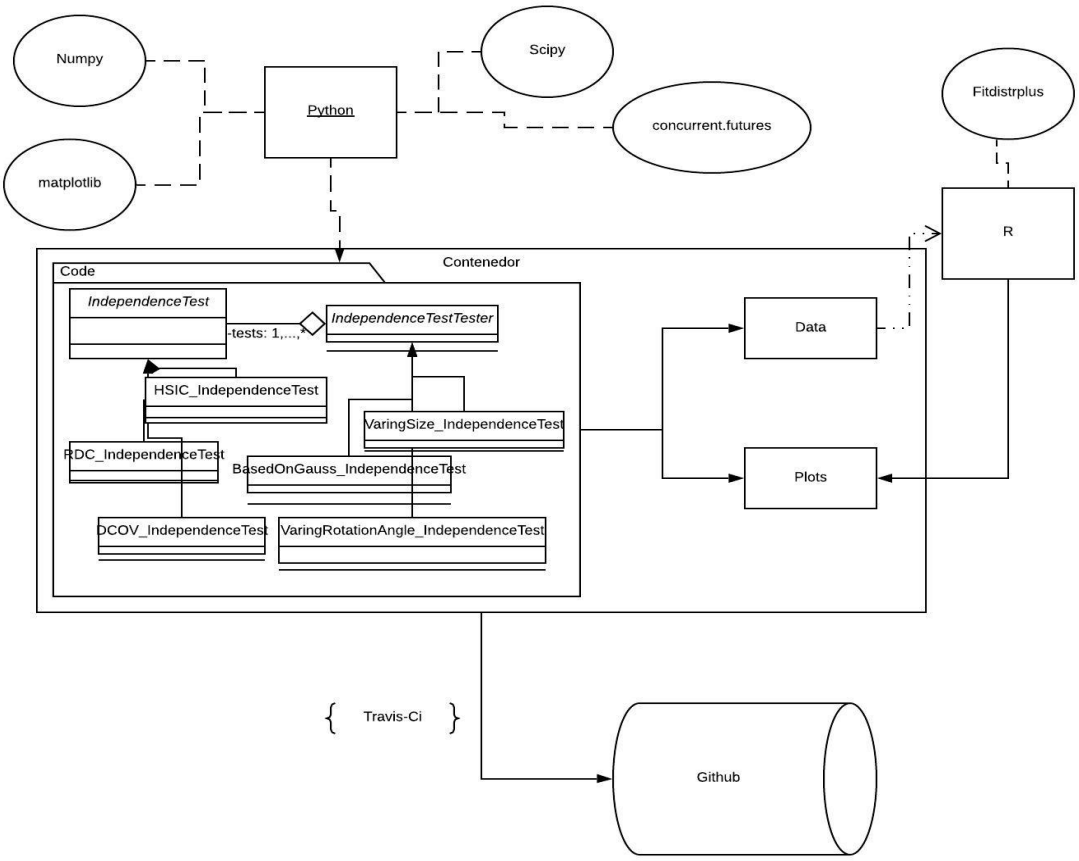


Figure 4.1: Diagram showcasing the different tools used to create this project and their relations.

4.4. Software development methodology

Previous section explains how we've implemented the code, in this section we will focus in the software development aspect of the project. Which life cycle was followed, how was the process flow and all related aspects of this matter.

For this whole project we followed an agile methodology, in which we met every week and discussed the problems which rose during the week. The main goal of agile methodology is to adapt to change and problems which may rise as the project develops, furthermore it's cyclic nature for a project like this where there are only two involved performs remarkably well.

We adapted the general aspects of agile methodology to our project for a perfect fit. For each cycle we always gathered information about the dependence measure, starting with RDC, followed with it's complete understanding with lots of meetings solving each misinterpretation or doubt which may had occurred and concluding with it's implementation in python. As the project grew new cycles for each dependence measure began and new problems rose in previous cycles which made us stepping back to a previous phase of the cycle. Finally once the implementation concluded and we dive completely into writing the degree work a new take on the development was chosen, where we worked in small cycles where there were small turn ins with concluded chapters and while a chapter was being double checked, another was being written.

Figure 4.2 summarizes how we adapted agile methodology for our project

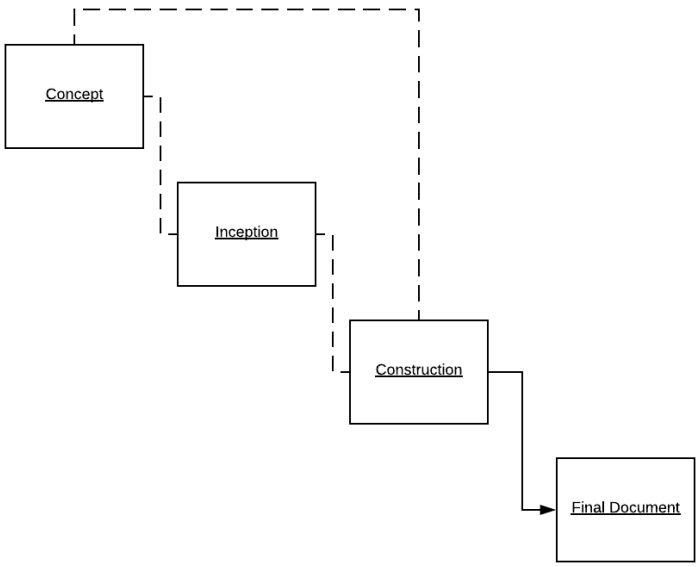


Figure 4.2: Diagram showcasing the software development methodology used.

5

Experiments

In this chapter we will present the results of various experiments in which we will compare the power of the explained tests between them and with other state-of-the-art independence tests, as well as comparing the power of these tests based on their asimptotic distribution and their empirical distribution. and [8]. This experiments will help us analyze in which conditions one dependence measure may be better than other. This is of key importance in various fields, such as multiple linear regression, where one of the hypothesis for the model to be functional is for all variables to be independent. Therefore finding if there is a dependent variable and erasing it from the model can drastically affect the performance of the classification. As our main goal is to clarify in which scenarios each dependence measure is optimal we will start measuring the power of each test for different data sets and modifications, then we will analyze it's asymptotic version for times when time may be of critical importance, and we will conclude measuring average times and comparing each test by complexity.

5.1. Power

In this section we will measure the power of each test, we will define power as the percentage of the times the null hypothesis's rejected for a given confidence level. In all our experiments, we set the number of random features for RDC to k = 3, and the random sampling width to s = 10^{-2}. All kernel methods make use of a Gaussian kernel with width hyper-parameter set to the median of the euclidean distances between samples of each of the input random variables.

First we will turn the issue of estimating the power of the RDC,HSIC and DCOV estimator. We define the power of a dependence measure as the percentage of times that it is able to discern between two samples with equal marginals, but one of them containing dependence.

In order to simulate the null hypothesis of our tests($\mathcal{H}_0$, the variables are independent) we will generate 500 samples under $\mathcal{H}_0$ to compute the threshold of the statistics with a signification level $\alpha = 0{,}05$. This will stand for our first group of experiments.

First we generated 500 pairs of 200 i.i.d. samples, in which the input variable was uniformly distributed on the unit interval, for each pair we generated each statistic, afterwards we calculated the 95

percentile, this will be the threshold for our test in this experiments.

To do so, we created three different experiments:

In the first one, adapted from [16], we studied 12 association patterns: linear, parabolic, quadratic, sin(4πx), sin(16πx), fourth root, circle, step, xsin(x),logarithm, Gaussian and a 2D multivariate normal distribution.Figure 5.1 shows graphically each association pattern.

Secondly for each of the 12 association patterns, we studied how Gaussian noise may affect the power of our test, with a noise increasing from 0 to 3 in 10 steps we generated 200 repetitions of 200 samples uniformly distributed on the unit interval and generated the pair with each association pattern, then we added Gaussian noise to the pair and normalized both marginals. Figure 5.2 shows for each subplot the power obtained with each association pattern. The x axis represents how the noise increases, and the y axis the power of the tests.

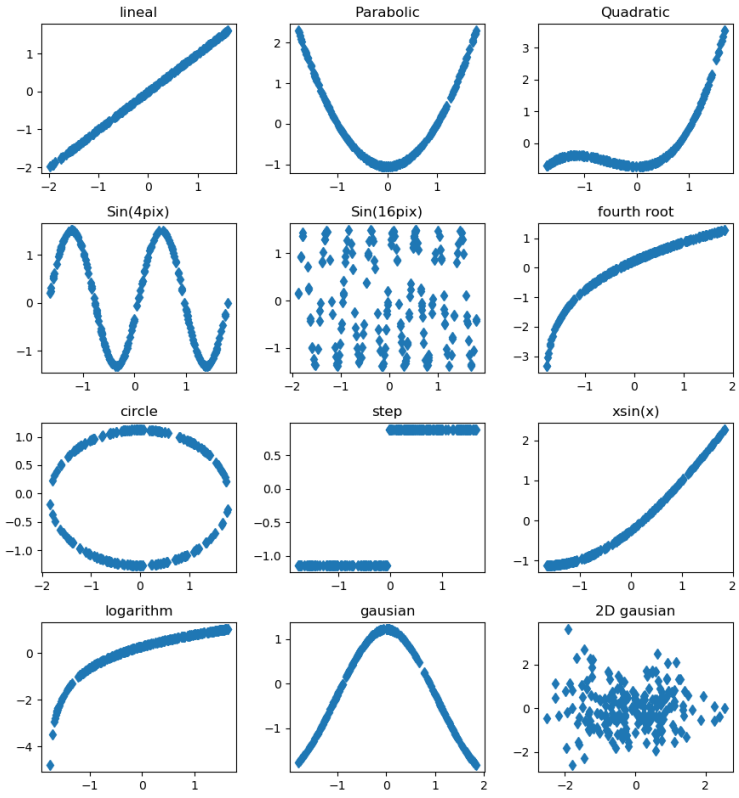


Figure 5.1: Representation of non linear dependence patterns

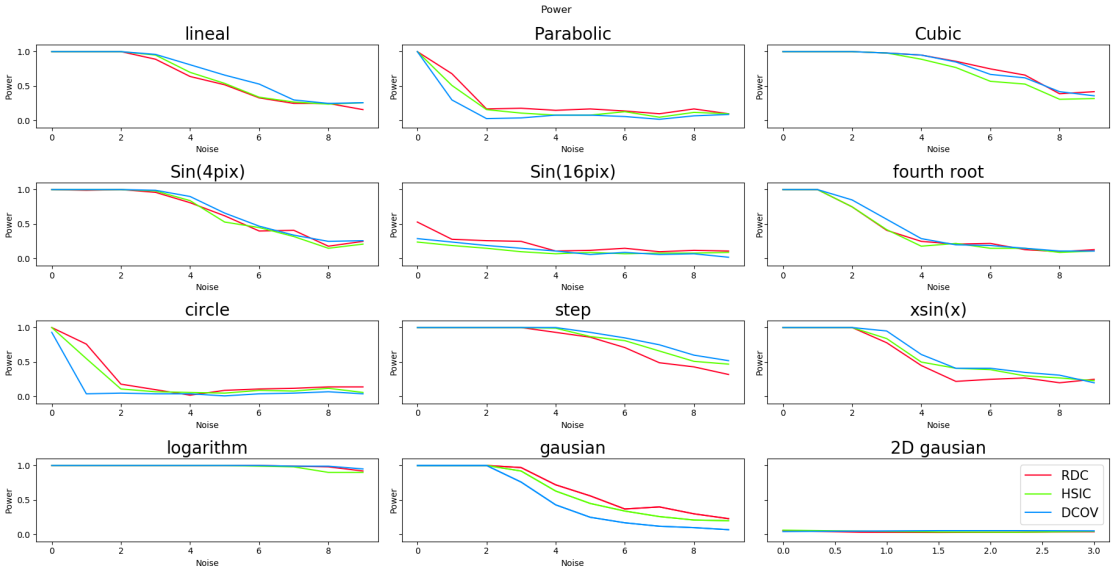


Figure 5.2: Power of tests adding Gaussian noise to marginals

In our second experiment we studied different sets of data and studied how the sample size affected the power of our tests. This test is taken from [20], the data sets are also taken from [20]. The first data set is a bivariate Gaussian with a correlation of 0.5, $(X, Y) \sim \mathcal{N}(0, \Sigma)$, where:

$$\Sigma = \begin{vmatrix} 1 & 0{,}5 \\ 0{,}5 & 1 \end{vmatrix}$$

For the second set we generated a uniform random variable $Z \sim U[0, 2]$. The marginals for this set will be constructed by:

$$X = ZX' \text{and} Y = ZY'$$

where $X', Y' \sim \mathcal{N}(0, 1)$, X' and Y' are independent, still X and Y are dependent due to both sharing the variable Z.

The variables X and Y in the third example are the marginals of a mixture of three bivariate Gaussians with correlations 0,0.8 and -0.8, with respective probabilities of 0.6, 0.2 and 0.2. The vector (X,Y) has density:

$$0{,}6\mathcal{N}(0, \Sigma_1) + 0{,}2\mathcal{N}(0, \Sigma_2) + 0{,}2\mathcal{N}(0, \Sigma_3)$$

Where

$$\Sigma_1 = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \Sigma_2 = \begin{vmatrix} 1 & 0{,}8 \\ 0{,}8 & 1 \end{vmatrix} \Sigma_2 = \begin{vmatrix} 1 & -0{,}8 \\ -0{,}8 & 1 \end{vmatrix}$$

The variables of the last example are generated as bivariate Gaussian random variable with correlation of 0.8 and then multiply each marginal with white Gaussian noise:

$$(X, Y) = (Z_1\epsilon_1, Z_2\epsilon_2) \text{ where } Z \sim \mathcal{N}(0, \Sigma_2) \text{ and } \epsilon_1, \epsilon_2 \sim \mathcal{N}(0, \Sigma_1)$$

Below samples from this data sets are displayed in 5.3. The power is measured for sample sizes 10, 91, 173, 255, 336, 418 and 500. For this experiment and the next one, we also compared the performance of RDC,HSIC and DCOV with other state of the art independence measures, being :

1.– Energy distance to compute the non-Gaussianity of the projections, ”Emean” and ”Emax” denote taking the mean and the maximum of the differences respectively.

2.– MMD, where ”MMDmean” and ”MMDmax” denote the methods where MMD are used instead of negentropy

3.– the non-Gaussianity test when we are taking the mean of the differences of the negentropy over ρ, denoted by "gaussmean".

The results of this experiment is presented in Figure 5.4.

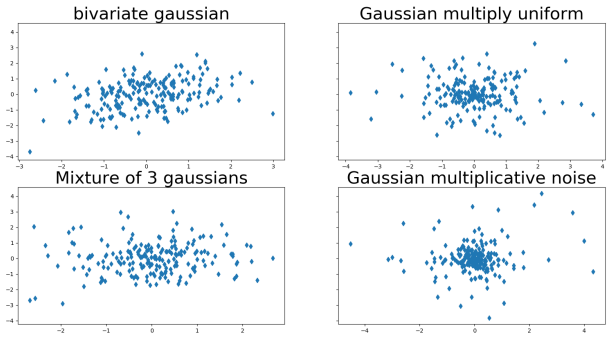


Figure 5.3: Samples from the data sets for the second experiment

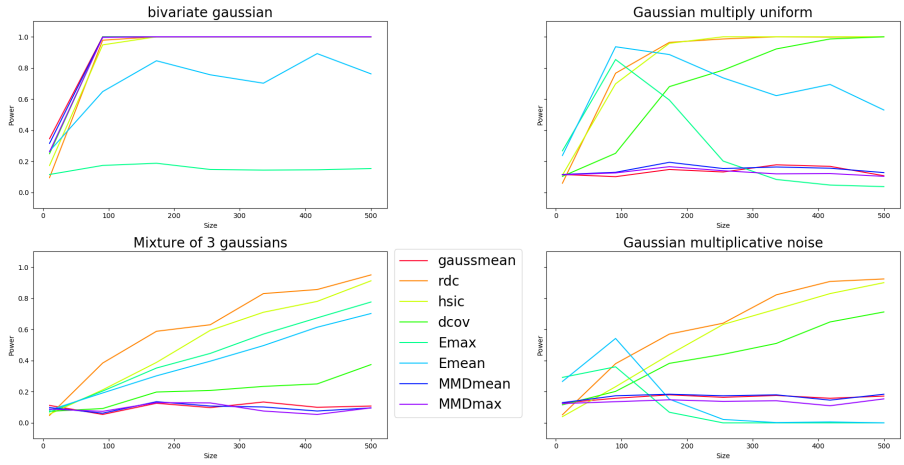


Figure 5.4: Power of tests adding Gaussian noise to marginals

For this set of experiments in which we try to determine the power of the tests, we have performed

a final experiment following [20] in which we studied the power of the tests and how they are affected by the rotation of the set. For this experiment we will use two independent random variables, X and Y, where X is a uniform random variable $(X \sim U[-\sqrt{3}, \sqrt{3}])$ whereas Y is a mixture of two uniform random variables, each having equal probability of occurrence on disjoint supports. That is, Y has density: $0{,}5U[-1, 0{,}5] + 0{,}5U[0{,}5, 1]$.

We generate new pairs of random variables by rotating this random pair (X,Y). This will affect the dependence between them, this variables will be independent if and only if the angle of rotation is an integer multiple of pi, $n \cdot \pi : n \in \mathbb{Z}$. After this rotation we had scaled X,Y to have zero mean and unit variance. For this experiment we have generated 500 samples and tested the power for 100 rotation angles going from 0 to 2π, with sample size 200. In Figure 5.5 shows samples of the same data with different rotation angle. As we can see in Figure 5.6 the function power for all tests is a $\frac{\pi}{2}$ even periodic function, which confirms that the potency of our tests does not depend on the sign of the correlation.

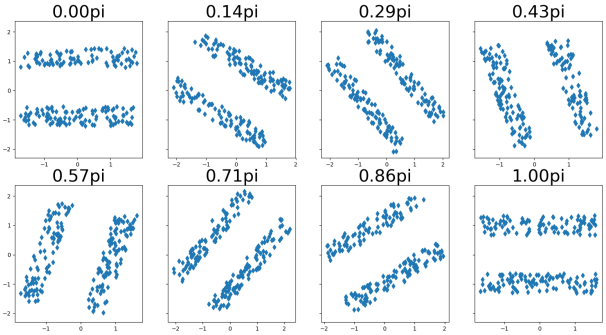


Figure 5.5: Samples from the data sets for the third experiment

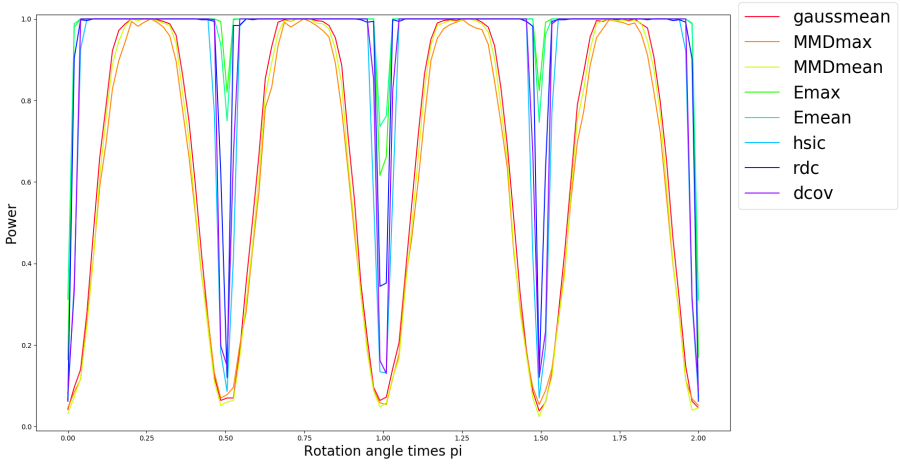


Figure 5.6: Power of the tests rotating the dataset

This concludes our first set of experiments, in the three experiments shown we can see that HSIC,DCOV and RDC are the sturdiest tests showing the best performance consistently. Among these three tests

RDC has proven to be the most consistent test, outperforming almost every time the other tests.

In Appendix C Section C.2 we reproduced this experiments for the asymptotic distribution.

5.2. Time complexity analysis

As we have seen in this experiments generally RDC outperforms the rest of tests, both in it's *real* and asymptotic version. Now to conclude this set of experiments, we will study the time complexity of each dependence measures.

We will start now studying the average complexity for each algorithm. Table 5.2 contains different charasteristics of the studied tests, as well as Pearson's ρ to compare. Taken from [16].

Coefficient	Non-Linear	N dimensional	Complexity
Pearson's ρ	✗	✗	O(n)
HSIC	✓	✓	O(n^2)
DCOV	✓	✓	O(n^2)
RDC	✓	✓	O($k^2 n$)

Table 5.1: Table with differences between the statistics, including average complexity of each algorithm.

Finally, we calculated the average time it takes to calculate each statistic for sample sizes going from 10 to 1000. In Figure 5.7 we showcase the results of this calculus.ç

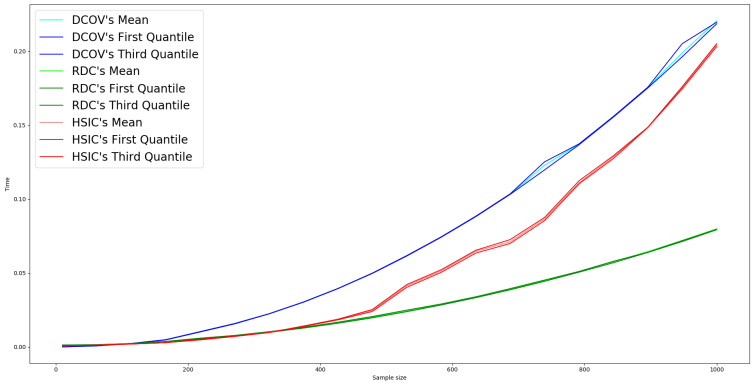


Figure 5.7: Comparison of time needed to calculate each statistic with different sample sizes, going from 10 to 1000

Coefficient	Quartiles	10	100	200	500	1000
HSIC	$Q1$	0,00063562	0,00194275	0,00474513	0,04035962	0,20322561
	Mean	0,00065339	0,0019827	0,00509361	0,04139885	0,20428318
	$Q3$	0,00066167	0,00196439	0,00520128	0,04222608	0,20520276
DCOV	$Q1$	0,00060457	0,00225528	0,01014042	0,06151939	0,21865168
	Mean	0,00063277	0,00234254	0,01022962	0,06166389	0,22060052
	$Q3$	0,00065571	0,0023856	0,01037216	0,06181291	0,21971462
RDC	$Q1$	0,00121409	0,00222874	0,00546312	0,0238595	0,07927161
	Mean	0,00136412	0,00228262	0,00583492	0,02420388	0,07994805
	$Q3$	0,00129074	0,00233924	0,00593042	0,02490777	0,07948118

Table 5.2: Table with the first, second and third quantile of the time to compute each statistic for sample sizes 10, 100, 200, 500 and 1000.

As we can see in Figure 5.7, RDC is considerably faster than HSIC and DCOV. In the figure we can see how around 500 the time curve changes slope, that is because for samples larger than 500 for a bivariate Gaussian the optimal k changes from 3 to 4, therefore the slope increases on the basis of $\frac{16}{9}$. Figure 5.8 showcases a polynomial approximation for the times, where we can see how HSIC follows a quadratic form with respect to the sample size , while RDC follows a linear form with respect to the sample size.

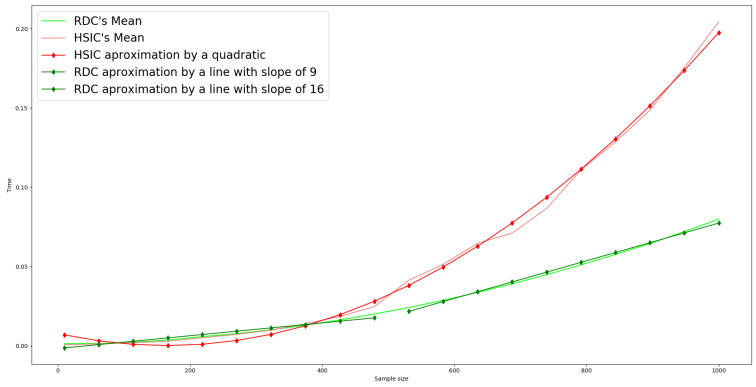


Figure 5.8: Theoretical polinomical approximation of HSIC and RDC put against the mean of the time needed to calculate the statistic

6

Conclusions and Future Work

For this project our goal is to present implement and compare different dependance measured. This, as mentioned douring the project, is of key importance for various search fields and techniques, such as PCA and multiple linear regression. Appendix C shows a simple example of how measuring dependence can drastically affect the performance of multiple linear regression.

The project starts presenting a homogeneity test based on embeddings of probability distributions into RKHS's, MMD, which allow us to express any probability distribution as an element of a Reproducing Kernel Hilbert Space (RKHS). As these spaces come with an associated distance, this homogeneity test consist in evaluating the distance between the two probability distributions.

From this homogeneity test, we define a independence test called HSIC applying the concepts learned from MMD. To perform an independence test one simply needs to compute the distance between the element in the RKHS that corresponds to the joint distribution of the random variables considered, and the element onto which the product of the marginal is transformed. Given that the joint distribution of two independent random variables is the product of the marginals, if this distance is non-zero, the variables are dependent. If the RKHS is sufficiently rich, a zero value of this distance also characterizes independence.

We follow the journey talking about Energy distance, and how we can define a independence test called DCOV. Which is a $\mathcal{L}_2$ distance between the respective characteristic functions.

Finally, the last dependence measure we define is RDC which is an aproximation of the HGR correlation coefficient defined in [3] which due to being the supremum over a infinite-dimensional space makes it unmanageable. Therefore, we will augment our sample by projecting it with random non-linear projections to a bigger space, where the non-linear dependencies will be transformed to linear ones, allowing us to calculate the the equivalent supremum in a smaller space ($\mathbb{R}$).

After, we presented the design of the software, analyzing the requirements and presenting the skeleton for the software developed. This is followed by explaining the development process, diving into the tools used, and the software development methodology which we followed for this project.

Concluding this work, we present the experiments performed to the tests, where we've seen how

in most case scenarios RDC outperforms the other statistics. There are few case scenarios where it may be better to use another test, the most relevant one being the relation pattern step. Where DCOV outperformed RDC. This may be important to notice because this relation pattern is equivalent of two variables X,Y, where Y = Heaviside(X). This relation pattern is of key importance in various scientific fields, such as differential equations, where it represents a signal which switches on at a specified time and stays switched on indefinitely.

Therefore, for general purposes, we can conclude that RDC will be the best answer, because it's more time efficient and generally performs better than the other tests. However, if there is previous knowledge of the relation pattern that the data may follow, then DCOV or HSIC may be a better solution.

For future work, it'd be interesting to study other dependence patterns and other distributions, such as distributions with a heavy tail like the Levy probability distribution and setting the dependence on the tail. This may be interesting for studies such as migration patterns where the tail of it's distribution movement is of key importance. Furthermore, it'd be interesting to compare more independence measures, such as the ones based on mutual information, correntropy, and non-Gaussianity in order to make this project as complete as possible.

Bibliography

[1] L. D. Broemeling, "An account of early statistical inference in arab cryptology," *The American Statistician*, vol. 65, no. 4, pp. 255–257, 2011.

[2] I. Schneider, *Jakob Bernoulli, Ars conjectandi (1713)*, pp. 88–104. 12 2005.

[3] H. Gebelein, "Das statistische problem der korrelation als variations- und eigenwertproblem und sein zusammenhang mit der ausgleichsrechnung," *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 21, pp. 364 – 379, 11 2006.

[4] M. Schuld and N. Killoran, "Quantum machine learning in feature hilbert spaces," *Physical Review Letters*, vol. 122, 03 2018.

[5] T. Hsing and R. Eubank, *Theoretical foundations of functional data analysis, with an introduction to linear operators*. Wiley series in probability and statistics, Hoboken, NJ: Wiley, 2015.

[6] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.

[7] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, pp. e49–e57, July 2006.

[8] A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola, "A kernel statistical test of independence," pp. 585–592, 2008.

[9] K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf, "Kernel measures of conditional dependence," pp. 489–496, 2008.

[10] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, ALT'05, (Berlin, Heidelberg), pp. 63–77, Springer-Verlag, 2005.

[11] D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu, "Equivalence of distance-based and rkhs-based statistics in hypothesis testing," *Ann. Statist.*, vol. 41, pp. 2263–2291, 10 2013.

[12] R. Serfling, *Approximation theorems of mathematical statistics*. Wiley series in probability and mathematical statistics : Probability and mathematical statistics, New York, NY [u.a.]: Wiley, [nachdr.] ed., 1980.

[13] M. L. R. Gabor J.Székely and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," 2007.

[14] G. J.Székely and M. L. Rizzo, "Brownian distance covariance," 2009.

[15] G. J. S ZÉKELY and N. K. B AKIROV, "Extremal probabilities for gaussian quadratic forms. probab. theory related fields," 2008.

[16] D. Lopez-Paz, P. Hennig, and B. Schölkopf, "The randomized dependence coefficient," 2013.

[17] D. Lopez-Paz and B. Schölkopf, "The randomized dependence coefficient," 2013.

[18] R. Johnson and D. Wichern, *Applied multivariate statistical analysis*. Upper Saddle River, NJ: Prentice Hall, 5. ed ed., 2002.

[19] P. M. Plancherel, *Contribution À L'Étude de la reprÉsentation D'une fonction arbitraire par des intÉgrales dÉfinies*. 1910.

[20] M. Rao, S. Seth, J. Xu, Y. Chen, H. D. Tagare, and J. C. Príncipe, "A test of independence based on a generalized correlation function," *Signal Processing*, vol. 91, no. 1, pp. 15–27, 2011.

[21] G. Strang, *Linear algebra and its applications*. Belmont, CA: Thomson, Brooks/Cole, 2006.

[22] B. Hunter, John K.; Nachtergaele, *Applied Analysis*. 2001.

Appendices

A

Proofs and in depth content for chapter 2

A.1. MMD

Mean embedding additional content

Definition A.1.1. ***Riesz representation***

If T is a bounded linear operator on a Hilbert space $\mathcal{H}$, then there exist some $g \in \mathcal{H}$ such that $\forall f \in \mathcal{H}$:

$$T(f) =< f, g >_{\mathcal{H}}$$

Lemma A.1.1. *Given a k(s,) semi positive definite, measurable and $\mathbb{E}\sqrt{k(X,X)} < \infty$, where X$\sim \mathbb{P}$ then $\mu_p \in \mathcal{H}$ exist and fulfills the next condition $\mathbb{E}f(X) =< f, \mu_p >$ for all f $\in \mathcal{H}$*

proof

Lets define the linear operator $T_{\mathbb{P}} f \equiv \mathbb{E}(\sqrt{k(X,X)}) < \infty \forall f \in \mathcal{H}$

$$\begin{aligned} |T_{\mathbb{P}} f| &= |\mathbb{E}(f(X))| \\ &\leq \mathbb{E}(|f(X)|) \\ &\text{Reproducing property of the kernel} \\ &= \mathbb{E}|< f, k(\cdot, X) >_{\mathcal{H}}| \\ &\text{Chauchy Schwarz inequality} \\ &\leq \|f\|_{\mathcal{H}} \cdot \mathbb{E}(\sqrt{k(X,X)})^{1/2} \\ &\text{The expectation under } \mathbb{P} \text{ of the kernel is bounded} \\ &< \infty \end{aligned} \tag{A.1}$$

Then using the Riesz representation theorem applied to T_p, there exist a $\mu_p \in \mathcal{H}$ such that $T_p f =< f, \mu_p >_{\mathcal{H}}$

Equivalence for the definitions of Mean embedding If $f \in \mathcal{H}$ and $\mu_{\mathbb{P}} \in \mathbb{R}\,\mathbb{E}(f(X)) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(x_n)$

Applying the Riesz representation theorem to represent $f(x_n)$

$\forall x_n$ then:

$$f(x_n) =< f, k(\cdot, x_n) >_{\mathcal{H}}$$

then

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} f(x_n) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} < f, k(\cdot, x_n >_{\mathcal{H}} =< f, \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} k(\cdot, x_n) >_{\mathcal{H}}$$

which leads to the final conclusion:

$\mu_{\mathbb{P}} \equiv \mathbb{E}_{X\sim\mathbb{P}}(k(t, X))\ t \in [0, T]$

Proof for new interpretation of MMD

$$\begin{aligned} MMD &\equiv \sup_{f\in\mathcal{H}\|f\|\leq 1} \{\mathbb{E}(f(x)) - \mathbb{E}(f(y))\} \\ &= \sup_{f\in\mathcal{H}\|f\|\leq 1} \{< f, \mu_{\mathbb{P}} > - < f, \mu_{\mathbb{Q}} >\} \\ &= \sup_{f\in\mathcal{H}\|f\|\leq 1} < f, (\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}) > \\ &\leq {}^{1} \sup_{f\in\mathcal{H}\|f\|\leq 1} \{\|f\|_{\mathcal{H}}, \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}\} \\ &\leq \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}. \end{aligned} \tag{A.2}$$

But on the other side, if we choose f as:

$$f = \frac{1}{\|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|}(\mu_{\mathbb{P}} - \mu_{\mathbb{Q}})$$

then we have:

$$\sup_{f\in\mathcal{H}\ \|f\|\leq 1} \{\|f\|_{\mathcal{H}}, \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}\} \geq \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}$$

therefore

$$MMD = \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}$$

Proof Proposition 2.2.3

$$
\begin{aligned}
MMD^2(\mathcal{F},\mathbb{P},\mathbb{Q}) &= \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|^2_{\mathcal{H}} \\
&=< \mu_{\mathbb{P}} - \mu_{\mathbb{Q}}, \mu_{\mathbb{P}} - \mu_{\mathbb{Q}} >_{\mathcal{H}} \\
&=< \mathbb{E}(k(\cdot,X)) - k(\cdot,Y)), \mathbb{E}(k(\cdot,X')) - k(\cdot,Y')) > \\
&= \mathbb{E}(< k(\cdot,X), k(\cdot,X') > + < k(\cdot,Y), k(\cdot,Y') > -2 < k(\cdot,X)k(\cdot,Y) >) \\
&\text{applying the reproductive property of the kernel.} \qquad \text{(A.3)} \\
&= \mathbb{E}(k(X,X') + k(Y,Y') - 2K(X,Y)) \\
&= \mathbb{E}(k(X,X')) + \mathbb{E}(k(Y,Y')) - 2\mathbb{E}(k(X,Y)) \\
&= \int\int k(s,t) \underbrace{d(\mathbb{P}-\mathbb{Q})(s)}_{\text{Signed Measure}} d(\mathbb{P}-\mathbb{Q})(t)
\end{aligned}
$$

A.1.1. Proving that MMD defines an homogeneity test

Now with the content we've already explained we will prove that MMD defines an homogeneity test, this is that it is a metric between probability distributions. The first definition of being characteristic is notation which may be helpful when reading other books and papers.

Definition A.1.2. ***Characteristic kernel***

A reproducing kernel k is a characteristic kernel if the induced γ_k is a metric.

Theorem A.1.2. *If X is a compact metric space, k is continuous and $\mathcal{H}$ is dense in $\mathbb{C}$(X) with respect to the supremum norm, then $\mathcal{H}$ is characteristic.*

proof

Being characteristic means that

$MMD(\mathcal{F}, \mathbb{P}, \mathbb{Q}) = 0 \leftrightarrow \mathbb{P} = \mathbb{Q}$

$\rightarrow$

By Lemma 2.2.1 we know that $\mathbb{P}$ and $\mathbb{Q}$ are equal if and only if $\mathbb{E}f(X) = \mathbb{E}f(Y)\ \forall f \in \mathcal{C}(\mathcal{X})$

Given that $\mathcal{H}$ is dense in $\mathcal{C}$(X) then:

$$\forall \epsilon > 0, f \in \mathcal{C}(X), \exists g \in \mathcal{H} : \|f - g\|_\infty < \epsilon$$

$$\begin{aligned}
|\mathbb{E}(f(X)) - \mathbb{E}(f(Y))| &= |\mathbb{E}(f(X)) - \mathbb{E}(g(X)) + \mathbb{E}(g(X)) - \mathbb{E}(g(Y)) + \mathbb{E}(g(Y)) - \mathbb{E}(f(Y))| \\
&\leq |\mathbb{E}(f(X)) - \mathbb{E}(g(X))| + |\mathbb{E}(g(X)) - \mathbb{E}(g(Y))| + |\mathbb{E}(g(Y)) - \mathbb{E}(f(Y))| \\
&= |\mathbb{E}(f(X)) - \mathbb{E}(g(X))| + |< g, \mu_\mathbb{P} - \mu_\mathbb{Q} >_\mathcal{H}| + |\mathbb{E}(g(Y)) - \mathbb{E}(f(Y))| \\
&\leq \mathbb{E}|f(X) - g(X)| + |< g, \mu_\mathbb{P} - \mu_\mathbb{Q} >_\mathcal{H}| + \mathbb{E}|g(Y) - f(Y)| \\
&\leq^1 \|f - g\|_\infty + |< g, \mu_\mathbb{P} - \mu_\mathbb{Q} >_\mathcal{H}| + \|f - g\|_\infty \\
&\leq |< g, \mu_\mathbb{P} - \mu_\mathbb{Q} >_\mathcal{H}| + 2\epsilon
\end{aligned} \tag{A.4}$$

By Lemma 2.2.3 we know that if MMD = 0 then $\mu_\mathbb{P} = \mu_\mathbb{Q}$. Hence:

$$|\mathbb{E}(f(X)) - \mathbb{E}(f(Y))| \leq 2\epsilon$$

Then by Lemma 2.2.1 $\mathbb{P}$ and $\mathbb{Q}$ are equal.

$\leftarrow$

By definition of MMD.

A.1.2. Tensor Products

Definition A.1.3. ***Tensor product*** *The tensor product of Hilbert spaces$\mathcal{H}_1$ and $\mathcal{H}_2$ with inner products $< , >_1$ and $< , >_2$ is defined as the completion of the space $\mathcal{H}_1 \times \mathcal{H}_2$ with inner product $< , >_1 < , >_2$extended by linearity. The resulting space is also a Hilbert space.*

Lemma A.1.3. *A kernel υ in the tensor product space $\mathcal{H} \times \mathcal{G}$ can be defined as:*

$$\upsilon((x, y), (x', y')) = k(x, x')l(y, y')$$

A.2. HSIC

A.2.1. HSIC in terms of the Cross Covariance

Definition A.2.1. ***Tensor product operator***

Let $h \in \mathcal{H}, g \in \mathcal{G}$. The tensor product operator $h \otimes g : \mathcal{G} \to \mathcal{H}$ is defined as:

$$(h \otimes g)(f) =< g, f >_{\mathcal{G}} h, \forall f \in \mathcal{G}$$

Definition A.2.2. ***Hilbert-Schmidt norm of a linear operator***

Let $C : \mathcal{G} \to \mathcal{H}$ be a linear operator between RKHS $\mathbb{G}$ and $\mathcal{H}$ the Hilbert-Schmidt norm of C is defined as:

$$\|C\| = \sqrt{\sum < Cv_j, u_i >^2_{\mathcal{H}}}$$

Where v_j and u_j are the orthonormal basis for $\mathbb{G}$ and $\mathcal{H}$ respectively.

Definition A.2.3. ***Cross-Covariance operator***

The cross-covariance operator associated with $\mathbb{P}_{XY}$ is the linear operator $C_{XY} : \mathcal{G} \to \mathcal{H}$ defined as:

$$C_{XY} = \mathbb{E}_{XY}[(\phi(X) - \mu_{\mathbb{P}}) \otimes (\psi(Y) - \mu_{\mathbb{Q}})] = \mathbb{E}_{XY}[\phi(X) \otimes \psi(Y)] - \mu_{\mathbb{P}} \otimes \mu_{\mathbb{Q}}$$

by applying the distributive property of the tensor product

Which is a generalisation of the cross-covariance matrix between random vectors.

Definition A.2.4. ***HSIC*** *We define the Hilbert-Schmidt Independence Criterion for $\mathbb{P}_{\mathcal{XY}}$ as the squared HS norm of the associated cross-covariance operator:*

$$HSIC(\mathbb{P}_{\mathcal{XY}}, \mathcal{H}, \mathcal{G}) = \|C_{XY}\|^2_{\mathcal{HS}} \tag{A.5}$$

A.2.2. demonstrations

Proof HSIC in terms of expectations 2.2 First we will simplify the notation of C_{XY}

$$C_{XY} = \mathbb{E}_{XY}[\phi(X) \otimes \psi(Y)] - \mu_{\mathbb{P}} \otimes \mu_{\mathbb{Q}} = \bar{C_{XY}} - M_{XY}$$

Using this notation:

$$\begin{aligned} \|C_{XY}\|^2_{\mathcal{HS}} &=< \bar{C}_{XY} - M_{XY}, \bar{C}_{X'Y'} - M_{X'Y'} >_{\mathcal{HS}} \\ &=< \bar{C}_{XY}, \bar{C}_{X'Y'} >_{\mathcal{HS}} + < M_{XY}, M_{X'Y'} > -2 < \bar{C}_{XY}, M_{X'Y'} >_{\mathcal{HS}} \end{aligned} \tag{A.6}$$

Now calculating each of this products individually:

$$\begin{aligned} < \bar{C}_{XY}, \bar{C}_{X'Y'} >_{\mathcal{HS}} &=< \mathbb{E}_{XY}[\phi(X) \otimes \psi(Y)], \mathbb{E}_{X'Y'}[\phi(X) \otimes \psi(Y)] > \\ &= \mathbb{E}_{XY}\mathbb{E}_{X'Y'}\|\phi(X) \otimes \psi(Y)\|^2 \\ &= \mathbb{E}_{XY}\mathbb{E}_{X'Y'}\|\phi(X)\|^2\|\psi(Y)\|^2 \\ &= \mathbb{E}_{XY}\mathbb{E}_{X'Y'} < \phi(X), \phi(X') >< \psi(Y), \psi(Y) > \\ &= \mathbb{E}_{XY}\mathbb{E}_{X'Y'}k(X, X')l(Y, Y') \end{aligned} \tag{A.7}$$

$$\begin{aligned} < M_{XY}, M_{X'Y'} >_{\mathcal{HS}} &=< \mu_{\mathbb{P}} \otimes \mu_{\mathbb{Q}}, \mu_{\mathbb{P}} \otimes \mu_{\mathbb{Q}} >_{\mathcal{HS}} \\ &= \|\mu_{\mathbb{P}} \otimes \mu_{\mathbb{Q}}\|^2_{\mathcal{HS}} \\ &= \|\mu_{\mathbb{P}}\|^2_{\mathcal{H}}\|\mu_{\mathbb{Q}}\|^2_{\mathcal{G}} \\ &=< \mu_{\mathbb{P}}, \mu_{\mathbb{P}} >_{\mathcal{H}} < \mu_{\mathbb{Q}}, \mu_{\mathbb{Q}} >_{\mathcal{G}} \\ &=< \mathbb{E}_X k(X, \cdot), \mathbb{E}_{X'} k(X', \cdot) >_{\mathcal{H}} < \mathbb{E}_Y l(Y, \cdot), \mathbb{E}_{Y'} k(Y', \cdot) >_{\mathcal{G}} \\ &= \mathbb{E}_X\mathbb{E}_{X'}\mathbb{E}_Y\mathbb{E}_{Y'} < k(X, \cdot), k(X', \cdot) >_{\mathcal{H}} < l(Y, \cdot), l(Y', \cdot) >_{\mathcal{G}} \\ &= \mathbb{E}_X\mathbb{E}_{X'}\mathbb{E}_Y\mathbb{E}_{Y'}k(X, X')l(Y, Y') \end{aligned} \tag{A.8}$$

$$\begin{aligned} < \bar{C}_{XY}, M_{XY} >_{\mathcal{HS}} &=< \mathbb{E}_{XY}[\phi(X) \otimes \psi(Y)], \mu_{\mathbb{P}} \otimes \mu_{\mathbb{Q}} >_{\mathcal{HS}} \\ &=< \mathbb{E}_{XY}[\phi(X) \otimes \psi(Y)], \mathbb{E}_{X'}\phi(X') \otimes \mathbb{E}_{Y'}\psi(Y') >_{\mathcal{HS}} \\ &=< \mathbb{E}_{XY} < \mathbb{E}_{X'} < \mathbb{E}_{Y'} < \phi(X) \otimes \psi(Y), \phi(X') \otimes \psi(Y') >_{\mathcal{HS}} \\ &=< \mathbb{E}_{XY} < \mathbb{E}_{X'} < \mathbb{E}_{Y'} < \phi(X), \phi(X') >_{\mathcal{H}} < \psi(Y), \psi(Y') >_{\mathcal{G}} \\ &=< \mathbb{E}_{XY} < \mathbb{E}_{X'} < \mathbb{E}_{Y'}k(X, X')l(Y, Y'). \end{aligned} \tag{A.9}$$

A.2.3. HSIC empirical convergence

Theorem A.2.1. *let $\mathbb{E}_Z$ denote the expectation taken over m independent copies(x_i, y_i drawn from $P_{\mathcal{XY}}$. Then:*

$$HSIC(\mathbb{P}_{\mathcal{XY}}, \mathcal{H}, \mathcal{G}) = \mathbb{E}_Z[HSIC(Z, \mathcal{H}, \mathcal{G})] + O(m^{-1})$$

Proof

By definition of H we can write:

$$\mathbf{tr}KHLH = \mathbf{tr}KL - 2m^{-1}\mathbf{1}^T KL\mathbf{1} + m^{-2}\mathbf{tr}K\mathbf{tr}L$$

where $\mathbf{1}$ is the vector of all ones.

Now we will expand each of the terms separately and take expectations with respect to Z.

- $\mathbb{E}_Z[\mathbf{tr}KL]$:

$$\mathbb{E}_Z[\sum_i K_{ii}L_{ii} + \sum_{(i,j)\in i_2^m} K_{ij}Lji] = O(m) + (m)_2\mathbb{E}_{XYX'Y'}[k(X,X')l(Y,Y')]$$

Normalising terms by $\frac{1}{(m-1)^2}$ yields the first term, since $\frac{m(m-1)}{(m-1)^2} = 1 + O(m^{-1})$.

- $\mathbb{E}_Z[\mathbf{1}^T KL\mathbf{1}]$:

$$\mathbb{E}_Z[\sum_i K_{ii}L_{ii} + \sum_{(i,j)\in i_2^m}(K_{ii}Lij + K_{ij}Ljj)] + \mathbb{E}_Z[\sum_{(i,j,r)\in i_3^m} K_{ij}Ljr]$$

$$= O(m^2) + (m)_3\mathbb{E}_{XY}[\mathbb{E}_{X'}[k(x,x')]\mathbb{E}_{Y'}[l(Y,Y')]]$$

Again, normalising terms by $\frac{2}{(m-1)^2}$ yields the second term. As before we used that $\frac{m(m-1)}{(m-1)^2} = 1 + O(m-1)$.

- $\mathbb{E}_Z[\mathbf{tr}K\mathbf{tr}L]$:

$$O(m^3) + \mathbb{E}_Z[\sum_{(i,j,q,r)\in i_4^m} K_{ij}Lqr] = O(m^3) + (m)_4\mathbb{E}_{XX'}[k(x,x')]\mathbb{E}_{YY'}[l(Y,Y')]$$

Normalisation by $\frac{1}{(m-1)^2}$ takes care of the last term, which completes the proof.

A.3. Energy Distance

Proof Proposition 2.4.1 We will start analysing the expectations of the right hand side. We will use that for any positive random variable $Z > 0$, $\mathbb{E}Z = \int_0^\infty \mathbb{P}(Z > z)dz$

$$\begin{aligned}
\mathbb{E}|X - Y| &= \int_0^\infty \mathbb{P}(|X - Y| > u)du \\
&= \int_0^\infty \mathbb{P}(X - Y > u)du + \int_0^\infty \mathbb{P}(X - Y < u)du \\
&= \int_0^\infty \int_{-\infty}^\infty \mathbb{P}(X - Y > u|Y = y)d\mathcal{G}(y)du + \int_0^\infty \int_{-\infty}^\infty \mathbb{P}(X - Y < u|X = x)d\mathcal{F}(x)(y)du \\
&= 2\int_{-\infty}^\infty \int_0^\infty \mathbb{P}(X - Y > u|Y = y)du\mathcal{G}(y) + \int_{-\infty}^\infty \int_0^\infty \mathbb{P}(X - Y < u|X = x)du\mathcal{F}(x) \\
&= \int_{-\infty}^\infty \int_0^\infty \mathbb{P}(X > u + y)du\mathcal{G}(y) + \int_{-\infty}^\infty \int_0^\infty \mathbb{P}(Y > u + x)du\mathcal{F}(x)
\end{aligned} \tag{A.10}$$

Now we use the change of variables z = u + y for the first integral, and w = u + x for the second one. Applying Fubini again:

$$\begin{aligned}
\mathbb{E}|X - Y| &= \int_{-\infty}^\infty \int_y^\infty \mathbb{P}(X > z)dz\mathcal{G}(y) + \int_{-\infty}^\infty \int_x^\infty \mathbb{P}(Y > w)dw\mathcal{F}(x) \\
&= \int_{-\infty}^\infty \mathbb{P}(X > z)dz \int_y^\infty \mathcal{G}(y) + \int_{-\infty}^\infty \mathbb{P}(Y > w)dw \int_x^\infty \mathcal{F}(x) \\
&= \int_{-\infty}^\infty \mathbb{P}(X > z)\mathbb{P}(Y < z)dz + \int_{-\infty}^\infty \mathbb{P}(Y > w)\mathbb{P}(X < w)dw \\
&= \int_{-\infty}^\infty [(1 - \mathcal{F}(z))\mathcal{G}(z) + (1 - \mathcal{G}(z))\mathcal{F}(z)]dz \\
&= -2\int_{-\infty}^\infty \mathcal{F}(z)\mathcal{G}(z)dz + \mathbb{E}|X| + \mathbb{E}|Y|
\end{aligned} \tag{A.11}$$

Taking $\mathcal{F} = \mathcal{G}$ in the previous development:

$$\mathbb{E}|X - X'| = -2\int_{-\infty}^\infty \mathcal{F}^2(z)dz + 2\mathbb{E}|X|$$

Equivalently for Y. Combining these partial results concludes the proof.

Proof Proposition 2.4.2 To prove this proposition we need the following lemma.

Lemma A.3.1. *$\forall x \in \mathbb{R}^d$ then:*

$$\int_{\mathbb{R}^d} \frac{1 - cos(tx)}{\|t\|_d^{d+1}} dt = c_d \|x\|_d$$

where tx is the inner product of t and x.

proof We will begin by applying the following transformation: $z_1 = \frac{tx}{\|x\|_d}$ followed by the following change of variables: $s = z\|x\|_d$

$$\begin{aligned}
\int_{\mathbb{R}^d} \frac{1 - cos(tx)}{\|t\|_d^{d+1}} dt &= \int_{\mathbb{R}^d} \frac{1 - cos(z\|x\|_d)}{\|z\|_d^{d+1}} dt \\
&= \int_{\mathbb{R}^d} \frac{1 - cos(s)}{\frac{\|s\|_d}{\|x\|_d}^{d+1} \|x\|_d^d} dt \\
&= \|x\|_d \int_{\mathbb{R}^d} \frac{1 - cos(s)}{\|s\|_d^{d+1}} ds \\
&= \|x\|_d \frac{\pi^{\frac{d+1}{2}}}{\Gamma(\frac{d+1}{2})}
\end{aligned} \tag{A.12}$$

proof 2.4.1 Let $\overline{\phi_{\mathbb{P}}(t)}$ denote the complex conjugate of the characteristic function.

$$\begin{aligned}
|\phi_{\mathbb{P}}(t) - \phi_{\mathbb{Q}}(t)|^2 = &= (\phi_{\mathbb{P}}(t) - \phi_{\mathbb{Q}}(t))\overline{(\phi_{\mathbb{P}}(t) - \phi_{\mathbb{Q}}(t))} \\
&= (\phi_{\mathbb{P}}(t) - \phi_{\mathbb{Q}}(t))(\overline{\phi_{\mathbb{P}}(t)} - \overline{\phi_{\mathbb{Q}}(t)}) \\
&= \phi_{\mathbb{P}}(t)\overline{\phi_{\mathbb{P}}(t)} - \phi_{\mathbb{P}}(t)\overline{\phi_{\mathbb{Q}}(t)} - \phi_{\mathbb{Q}}(t)\overline{\phi_{\mathbb{P}}(t)} + \phi_{\mathbb{Q}}(t)\overline{\phi_{\mathbb{Q}}(t)} \\
&= \mathbb{E}[e^{itX}e^{-itX'}] - \mathbb{E}[e^{itX}e^{-itY}] - \mathbb{E}[e^{itY}e^{-itX}] + \mathbb{E}[e^{itY}e^{-itY'}] \\
&= \mathbb{E}[e^{it(X-X')} - e^{it(Y-X)} - e^{it(X-Y)} + e^{it(Y-Y')} \\
&= \mathbb{E}[cos(t(X - X')) + isin(t(X - X')) - cos(t(Y - X)) - isin(t(Y - X)) - cos(t(X - Y)) \\
&- isin(t(X - Y)) + cos(t(Y - Y')) + isin(t(Y - Y')) \\
&\text{sin(X) = -sin(-X), cos(X) = cos(-X), sin (x- y)=sin(x)cos(y)- cos(x)sin(y)} \\
&= \mathbb{E}[cos(t(X - X')) - 2cos(t(Y - X)) + cos(t(Y - Y')) \\
&+ isin(t(X - X')) + isin(t(Y - Y')) \\
&= \mathbb{E}[2(1 - cos(t(Y - X))) - (1 - cos(t(X - X'))) - (1 - cos(t(y - Y')))]
\end{aligned} \tag{A.13}$$

Applying Fubini and the previous lemma:

$$\begin{aligned}
\int_{\mathbb{R}^d} \frac{|\phi_{\mathbb{P}}(t) - \phi_{\mathbb{Q}}(t)|^2}{\|t\|_d^{d+1}} &= \int_{\mathbb{R}^d} \frac{\mathbb{E}[2(1 - cos(t(Y - X))) - (1 - cos(t(X - X')))]}{\|t\|_d^{d+1}} dt \\
&= 2\mathbb{E}[\int_{\mathbb{R}^d} \frac{1 - cos(t(Y - X))}{\|t\|_d^{d+1}}] - \mathbb{E}[\int_{\mathbb{R}^d} \frac{1 - cos(t(X - X'))}{\|t\|_d^{d+1}} - -\mathbb{E}[\int_{\mathbb{R}^d} \frac{1 - cos(t(Y - Y'))}{\|t\|_d^{d+1}} \\
&= 2\mathbb{E}[c_d\|Y - X\|] - \mathbb{E}[c_d\|X - X'\|] - \mathbb{E}[c_d\|Y - Y'\|] \\
&= c_d(2\mathbb{E}[\|Y - X\|] - \mathbb{E}[\|X - X'\|] - \mathbb{E}[\|Y - Y'\|]) \\
&= c_d\varepsilon(X, Y)
\end{aligned} \tag{A.14}$$

A.3.1. DCOV Convergence of the statistic

Now we will proove that this statistics converge almost surely when the random vectors have finite first moments.

Theorem A.3.2. *if* $\mathbb{E}\|X\| + \mathbb{E}\|Y\| < \infty$ *then*

$$\lim_{n\to\infty} \nu_n^2(x,y) \xrightarrow{a.s} \nu^2(X,Y)$$

In order to proove this theorem we will give an alternative definition of the empirical DCOV statistic in order to make an elegant demonstration.

Definition A.3.1. *Given all the introduction of this section it'd have been natural, but less elementary, to define* $\nu_n(x,y)$ *as* $\|f_{XY}^n(t,s) - f_X^n(t)f_Y^n(s)\|$ *where:*

$$f_{XY}^n(t,s) = \frac{1}{n}\sum_{k=1}^{n} exp[i < t, x_k > + i < s, y_k >]$$

is the empirical characteristic function of the sample $((x_1,y_1),...,(x_n,y_n))$ *and*

$$f_X^n(t) = \frac{1}{n}\sum_{k=1}^{n} exp[i < t, x_k >]$$

$$f_Y^n(s) = \frac{1}{n}\sum_{k=1}^{n} exp[i < s, y_k >]$$

are the marginal empirical characteristic functions of the X sample and Y sample, respectively.

The next theorem shows that the two definitions are equivalent.

Theorem A.3.3. *If (X,Y) is a sample from the joint distribution of (X,Y), then*

$$\nu_n^2(X,Y) = \|f_{XY}^n(t,s) - f_X^n(t)f_Y^n(s)\|^2$$

Proof Lemma A.3.1 implies that there exist constants c_p and c_q such that for all $X \in \mathbb{R}^p, y \in \mathbb{R}^q$.

$$\begin{aligned}
&\int_{\mathbb{R}^p} \frac{1 - exp[i < t, X >]}{\|t\|_p^{1+p}} dt = c_p \|X\|_p \\
&\int_{\mathbb{R}^q} \frac{1 - exp[i < s, Y >]}{\|s\|_p^{1+p}} dt = c_q \|Y\|_q \\
&\int_{\mathbb{R}^p}\int_{\mathbb{R}^q} \frac{1 - exp[i < t, X > + i < s, Y >]}{\|t\|_p^{1+p}\|s\|_p^{1+p}} dt = c_q c_p \|X\|_p \|Y\|_q
\end{aligned} \tag{A.15}$$

where the integrals are understood in the principal value sense. For simplicity, consider the case p=q=1. The distance between the empirical characteristic functions in the weighted norm involves

$\|f^n_{XY}(t,s)\|^2$, $\|f^n_X(t)f^n_Y(s)\|^2$ and $\overline{f^n_{XY}(t,s)}f^n_X(t)f^n_Y(s)$. Now we will give the result of evaluating this, due to the similarity to previous demostrations.

$\|f^n_{XY}(t,s)\|^2 = \frac{1}{n^2}\sum_{k,l=1}^n cos(X_k - X_l)tcos(Y_k - Y_l)s + V_1$

where V_1 represents terms that vanish when the integral $\|f^n_{XY}(t,s) - f^n_X(t)f^n_Y(s)\|^2$ is evaluated.

$\|f^n_X(t)f^n_Y(s)\|^2 = \frac{1}{n^2}\sum_{k,l=1}^n cos(X_k - X_l)t + \frac{1}{n^2}\sum_{k,l=1}^n cos(Y_k - Y_l)s + V_2$

$\overline{f^n_{XY}(t,s)}f^n_X(t)f^n_Y(s) = \frac{1}{n^3}\sum_{k,l,m=1}^n cos(X_k - X_l)tcos(Y_k - Y_l)s + V_3$

where V_2 and V_3 represent terms that vanish when the integral is evaluated. To evaluate the integral $\|f^n_{XY}(t,s) - f^n_X(t)f^n_Y(s)\|^2$, apply Lemma A.3.1 and use:

$$cos(u)cos(v) = 1 - (1 - cos(u)) - (1 - cos(v)) + (1 - cos(u))(1 - cos(v))$$

After cancellation in the numerator of the integrand it remains to evaluate integrals of the type:

$$\begin{aligned}\int_{\mathbb{R}^2}(1 - cos(X_k - X_l)t)(1 - cos(Y_k - Y_l)s)\frac{dt}{t^2}\frac{ds}{s^2} &= \int_{\mathbb{R}}(1 - cos(X_k - X_l)t)\frac{dt}{t^2}\int_{\mathbb{R}}(1 - cos(Y_k - Y_l)s)\frac{ds}{s^2} \\ &= c_1^2\|X_i - X_j\|\|Y_i - Y_j\|\end{aligned} \tag{A.16}$$

where the first equality comes from applying Fubini.

For random vectors $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}^q$, the same steps are applied.Thus

$$\|f^n_{XY}(t,s) - f^n_X(t)f^n_Y(s)\|^2 = S_1 + S_2 - 2S_3$$

Where:

$$\begin{aligned}S_1 &= \frac{1}{n^2}\sum_{i,j=1}^n \|x_i - x_j\|_p\|y_i - y_j\|_q \\ S_2 &= \frac{1}{n^2}\sum_{i,j=1}^n \|x_i - x_j\|_p \frac{1}{n^2}\sum_{i,j=1}^n \|x_i - x_j\|_p\|y_i - y_j\|_q \\ S_1 &= \frac{1}{n^3}\sum_{i=1}^n\sum_{j,k=1}^n \|x_i - x_j\|_p\|y_i - y_k\|_q\end{aligned} \tag{A.17}$$

Now that we have proven the equality we will proove the theorem A.3.2 **proof** Define

$$\zeta_n(t,s) = \frac{1}{n}\sum_{k=1}^n e^{i<t,X_k>+i<s,Y_k>} - \frac{1}{n}\sum_{k=1}^n e^{i<t,X_k>}\frac{1}{n}\sum_{k=1}^n e^{i<s,Y_k>}$$

so that $\nu_n^2 = \|\zeta_n(t,s)\|^2$. Then after elementary transformations: $u_k = exp(i < t, X_k >) - f_X(t)$

and $v_k = exp(i < s, Y_k >) - f_Y(s)$.

For each $\theta > 0$ define the region:

$$D(\theta) = \{(t,s) : \theta \leq \|t\|_p \leq \frac{1}{\theta}, \theta \leq \|s\|_q \leq \frac{1}{\theta}\}$$

and random variables

$$\nu^2_{n,\theta} = \int_{D(\theta)} \|\zeta_n(t,s)\|^2 dw$$

For any fixed $\theta > 0$, the weight function w(t,s) is bounded on $D(\theta)$. Hence $\nu^2_{n,\theta}$ is a combination of V-statistics of bounded random variables, therefore by the strong law of large numbers it follows almost surely.

$$\lim_{n\to\infty} \nu^2_{n,\theta} = \nu^2_{\cdot,\theta} = \|f_{XY}(t,s) - f_X(t) f_Y(s)\|^2 dw$$

Clearly $\nu^2_{\cdot,\theta}$ converges to ν^2 as θ tends to zero. Now it remains to prove that almost surely

$$\lim \sup_{\theta\to 0} \lim \sup_{n\to\infty} \|\nu^2_{n,\theta} - \nu^2_n\| = 0$$

For each $\theta > 0$

$$\begin{aligned} \|\nu^2_{n,\theta} - \nu^2_n\| \leq & \int_{\|t\|_p \leq \theta} \|\zeta(t,s)\|^2 dw + \int_{\|t\|_p > \frac{1}{\theta}} \|\zeta(t,s)\|^2 dw \\ & + \int_{\|s\|_q \leq \theta} \|\zeta(t,s)\|^2 dw + \int_{\|s\|_q > \frac{1}{\theta}} \|\zeta(t,s)\|^2 dw \end{aligned} \tag{A.18}$$

For z = $(z_1, .., z_p)$ in $\mathbb{R}^p$ define the function

$$G(y) = \int_{\|z\|<y} \frac{1 - cos(z_1)}{\|z\|^{1+p}}$$

Clearly G(y) is bounded by c_p and $\lim_{y\to 0} G(y) = 0$. Applying the inequality $\|x + y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ and the following inequality.

Proposition A.3.4. *The Cauchy–Schwarz inequality states that for all vectors u and v of an inner product space it is true that*

$|\langle \mathbf{u}, \mathbf{v} \rangle|^2 \leq \langle \mathbf{u}, \mathbf{u} \rangle \cdot \langle \mathbf{v}, \mathbf{v} \rangle$

where $\langle \cdot, \cdot \rangle$ is the inner product. By taking the square root of both sides, and referring to the norms of the vectors, the inequality is written as [21], [22]

$|\langle \mathbf{u}, \mathbf{v} \rangle| \leq \|\mathbf{u}\| \|\mathbf{v}\|$

If $u_1, \ldots, u_n \in \mathbb{C}$ and $v_1, \ldots, v_n \in \mathbb{C}$, and the inner product is the standard complex inner product, then the inequality may be restated more explicitly as follows

$$|u_1\bar{v}_1 + \cdots + u_n\bar{v}_n|^2 \leq (|u_1|^2 + \cdots + |u_n|^2)(|v_1|^2 + \cdots + |v_n|^2)$$

one can obtain that:

$$\begin{aligned} \|\zeta_n(t,s)\|^2 &\leq 2\|\frac{1}{n}\sum_{k=1}^{n} u_k v_k\|^2 + 2\|\frac{1}{n}\sum_{k=1}^{n} u_k \frac{1}{n}\sum_{k=1}^{n} v_k\|^2 \\ &\leq \frac{4}{n}\sum_{k=1}^{n}\|u_k\|^2 \frac{1}{n}\sum_{k=1}^{n}\|v_k\|^2 \end{aligned} \tag{A.19}$$

Therefore the first summand in A.3.1 satisfies

$$\int_{\|t\|_p \leq \theta} \|\zeta(t,s)\|^2 dw \leq \frac{4}{n}\sum_{k=1}^{n} \int_{\|t\|_p \leq \theta} \frac{\|u_k\|^2 dt}{c_p \|t\|_p^{1+p}} \frac{1}{n}\sum_{k=1}^{n} \int_{\mathbb{R}^q} \frac{\|v_k\|^2 ds}{c_q \|s\|_q^{1+q}}$$

Here $\|v_k\|^2 = 1 + \|f_Y(s)\|^2 - exp(i < s, Y_k >)\overline{f_Y(s)} - exp(-i < s, Y_k >)f_Y(s)$, thus

$$\int_{\mathbb{R}^q} \frac{\|v_k\|^2 ds}{c_q \|s\|_q^{1+q}} = (2E_Y\|Y_k - Y\| - E\|Y - Y'\|) \leq 2(\|Y_k\| + E\|Y\|)$$

where the expectation E_Y is taken with respect to Y, and Y'$=^D$ Y is independent of Y_k. Further, after a suitable change of variables

$$\begin{aligned} \int_{\|t\|_p \leq \theta} \frac{\|u_k\|^2 dt}{c_p \|t\|_p^{1+p}} &= 2E_X\|X_k - X\|G(\|X_k - X\|\theta) - E\|X - X'\|G(\|X - X'\|\theta) \\ &\leq 2E_X\|X_k - X\|G(\|X_k - X\|\theta) \end{aligned} \tag{A.20}$$

Therefore from A.3.1:

$$\|\zeta_n(t,s)\|^2 \leq 4\frac{2}{n}\sum_{k=1}^{n}(|Y_k| + E|Y|)\frac{2}{n}\sum_{k=1}^{n} E_X\|X_k - X\|G(\|X_k - X\|\theta)$$

By the SLLN:

$$\limsup_{n\to\infty} \int_{\|t\|_p \leq \theta} \|\zeta_n(t,s)\|^2 dw = 0$$

almost surely.

Now for the second summand in A.3.1. Inequalities imply that $\|u_k\|^2 \leq 4$ and $\frac{1}{n}\sum_{k=1}^{n}\|u_k\|^2 \leq 4$.

Which lead us with further calculation to:

$$\limsup_{\theta\to 0} \limsup_{n\to\infty} \int_{\|t\|_p \leq \theta} \|\zeta_n(t,s)\|^2 dw = 0$$

For the rest of the summands it's pretty similar, which lead us to what we wanted to proof:

$$\lim_{\theta \to 0} \sup \lim_{n \to \infty} \sup \|\nu_{n,\theta}^2 - \nu_n^2\| = 0$$

Proposition A.3.5. *As a corollary for the theorem A.3.2 if* $\mathbb{E}\|X\| + \mathbb{E}\|Y\| < \infty$ *then*

$$\lim_{n \to \infty} \mathcal{R}_n^2(x, y) \xrightarrow{a.s} \mathcal{R}^2(X, Y)$$

A.4. RDC

Proof for Definition 2.5.2 In order to give consistency to the previous definition we need to introduce few mathematicall conceps:

Theorem A.4.1. *Probability Integral Transform Consider a random vector X = ($X_1, ..., X_d$) with continuous marginal cumulative distri- bution functions (cdfs) F_i, 1≤i≤d. **U**=($U_1, ..., U_d$):=($F_1(X_1)$,...,$F_d(X_d)$) has uniform marginals.*

proof Let X with CDF F and Y = F(X), then Y follows a uniform distribution. $F_Y(y) = P(Y \leq y) = P(F_X(x) \leq y) = P(X \leq F_X^{-1}(y)) = F_X(F_X^{-1}(y)) = y$

Proof for Theorem 2.5.2

$F_n := \frac{1}{n}\sum_{i=1}^{n} I(X_i \leq x)$ converges uniformly to P:

Let X = $(x_0, ..., x_m)$ such that $-\infty = x_0 < x_1 < ... < x_m = \infty$ and $F(x_j) - F(x_{j-1}) \leq \frac{1}{m}$ $F_n(x) - F(x) \leq F_n(x_j) - F(x_{j-1}) = F_n(x_j) - F(x_j) + \frac{1}{m}$

$F_n(x) - F(x) \geq F_n(x_{j-1}) - F(x_j) = F_n(x_{j-1}) - F(x_{j-1}) - \frac{1}{m}$

$$\|F_n - F\|_\infty = \sup_{x \in R} |F_n(x) - F(x)| \leq \max_{j \in 1,..,m} |F_n(x_j) - F(x_j)| + \frac{1}{m} \to 0(a.s)$$

B

Development further content

B.1. Gaussian Kernel

As we have seen a positive definite kernel $k(x, y)$ defines an inner product $k(x, y) =< \phi(x), \phi(y) >_{\mathcal{H}}$ for feature vector ϕ constructed from the input x, and $\mathcal{H}$ is a Hilbert space. The notation $< \phi(x), \phi(y) >$ means the inner product between $\phi(x)$ and $\phi(y)$. For a better understanding, you can imagine $\mathcal{H}$ to be the usual Euclidean space, but with an infinite number of dimensions. Then take a vector which is infinitely long, like $\phi(x) = (\phi_1(x), \phi_2(x), ...)$. In kernel methods, $\mathcal{H}$ is a RKHS (explained in the state of the art chapter 2, section 2.1). Since we only care about the inner product of the features, we will directly evaluate the kernel k. To explain smoothness of the functions given by the Gaussian Kernel, let us consider Fourier features. As it's easy to prove, $k(x, y) = k(x - y)$, the kernel only depends on the difference of the two arguments. Let $\hat{k}$ denote the Fourier transform of k.

In this Fourier viewpoint, the features of f are given by $f = (..., \frac{\hat{f}_l}{\sqrt{\hat{k}_l}}, ...)$, this is saying that the feature representation of your function f is given by its Fourier transform divided by the Fourier transform of the kernel k. The feature representation of x, which is $\phi(x)$ is:$(..., \sqrt{\hat{k}_l} \exp(-ilx), ...)$ where $i = \sqrt{-1}$. One can show that the reproducing property holds.

Now thanks to Plancherel theorem: [19]

It states that the integral of a function's squared modulus is equal to the integral of the squared modulus of its frequency spectrum. That is, if $f(x)$ is a function on the real line, and $\widehat{f}(\xi)$ is its frequency spectrum, then ;

$$\int_{-\infty}^{\infty} |f(x)|^2 \, dx = \int_{-\infty}^{\infty} |\widehat{f}(\xi)|^2 \, d\xi$$

Hence:

$$\|f\|_H^2 = \sum_{l=-\infty}^{\infty} \frac{\hat{f}_l^{\,2}}{\hat{k}_l}$$

Which as $f \in \mathcal{L}^2$ the norm is finite, the sum converges. Now as the Fourier transform of a Gaussian

kernel $K(x,y) = exp(-\frac{\|x-y\|^2}{\mu^2})$ is another Gaussian where $\hat{k_l}$ decreases exponentially fast with l. So if f is to be in this space, its Fourier transform must drop even faster than that of k. This means the function will have only a few low frequency components with high weights.(A function with only low frequency components is smooth).

B.2. Choosing Max Workers parameter

In order to showcase how the time efficiency drops as the number of max workers increases we designed a small experiment where we generated different thread pools with different maximum number of workers, and sequerntially made the pools work in a simple task, calculate the inverse of 40 different matrix , and we studied how the number of maximum workers affected the time efficiency of the task. Figure B.1 shows how at 8 workers reaches it's peak, which is the number of cores of the computer performed the experiment.

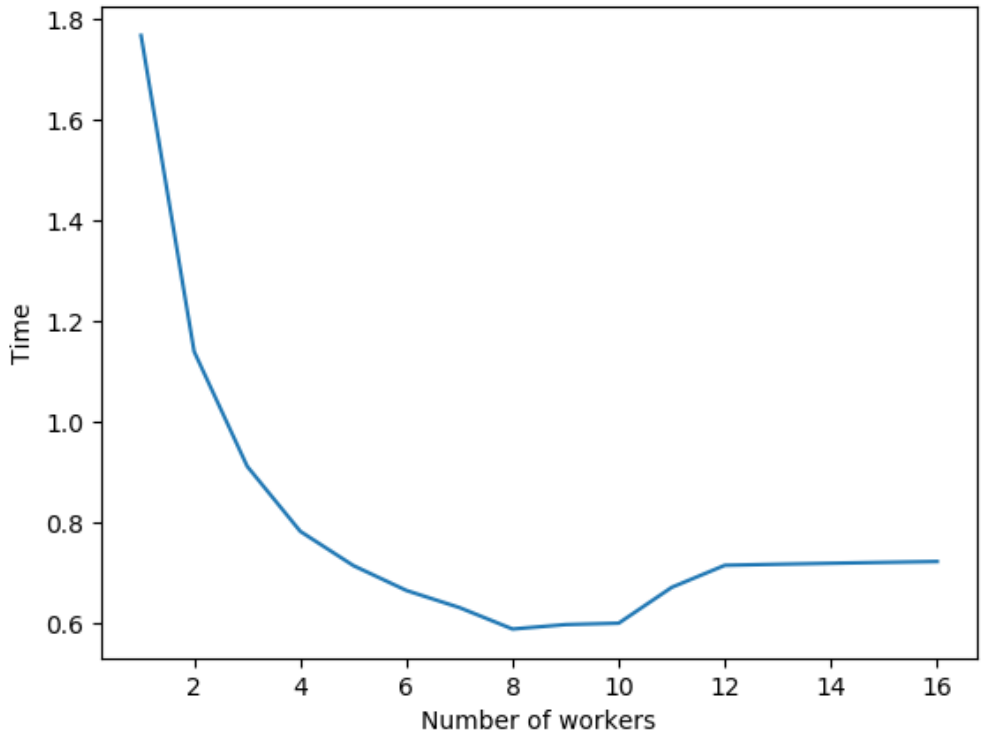


Figure B.1: Testing how the number of workers affects the performance of a task.

B.3. CANCOR CODE

Code B.1: Canonical correlation analysis in python

```
def cancor(x,y,k):
    canonical_correlation_matrix = np.cov(np.hstack([x, y]).T)

    k0 = k
    lower_bound = 1 #minimum k
    upper_bound = k #maximum k
    while True:
        #Canonical correlation
        k = int(k)

        C_XX = canonical_correlation_matrix[:k,:k]
        C_YY = canonical_correlation_matrix[k0:k0+k, k0:k0+k]
        C_XY = canonical_correlation_matrix[:k, k0:k0+k]
        C_YX = canonical_correlation_matrix[k0:k0+k, :k]

        eigs = np.linalg.eigvals(np.dot(np.dot(np.linalg.inv(C_XX), C_XY),
                                        np.dot(np.linalg.inv(C_YY), C_YX)))

        #Search if K is too large
        if not (np.all(np.isreal(eigs)) and
                0 <= np.min(eigs) and
                np.max(eigs) <= 1): #Condition of being too large
            upper_bound -= 1 #reduce the maximum in 1
            k = (upper_bound + lower_bound) / 2 #search in the middle
            continue

        if lower_bound == upper_bound: break #if lower_bound == upper_bound means we found the
            optimal value for k

        lower_bound = k #as k meets the condition we set the lower bound to k

        #Set k as the middle point
        if upper_bound == lower_bound + 1:
            k = upper_bound

        else:
            k = (upper_bound + lower_bound) / 2

    return np.sqrt(eigs),k
```

C

Experiments further content

C.1. Other experiments

Simple example of Multiple linear regression

We generated a dataset of X1,X2,X3,X4 and Y, Y is a linear combination of X1 and X2, and X3 and X4 are linear combinations of X1 and X2. We created the linear regression with r for two models: $Y \sim X1 + X2 + X3 + X4$ and $Y \sim X3 + X4$ and we obtained: We can see how for the second

Call: lm(formula = y x1 + x2 + x3 + x4, data = datos)
Residuals: Min 1Q Median 3Q Max -59.410 -5.069 -0.593 6.611 36.919
*Coefficients: Estimate Std. Error t value Pr(>|t|) (Intercept) 7.08261 3.60358 1.965 0.05228 . x1 0.63566 1.22597 0.518 0.60532 x2 0.00807 0.58582 0.014 0.98904 x3 -1.05341 0.51460 -2.047 0.04341 * x4 -0.43967 0.15336 -2.867 0.00511 ** — Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1*
Residual standard error: 13.11 on 95 degrees of freedom Multiple R-squared: 0.12, Adjusted R-squared: 0.0829 F-statistic: 3.237 on 4 and 95 DF, p-value: 0.01549

Frame C.1

model with only two variables it's not only easier to work with but it produces a better approximation.

Call: lm(formula = y x3 + x4, data = datos)
Residuals: Min 1Q Median 3Q Max -60.692 -5.776 -0.497 6.587 37.667
*Coefficients: Estimate Std. Error t value Pr(>|t|) (Intercept) 6.9817 2.8068 2.487 0.01457 * x3 -1.0342 0.5013 -2.063 0.04177 * x4 -0.4332 0.1509 -2.870 0.00503 ** — Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1*
Residual standard error: 12.99 on 97 degrees of freedom Multiple R-squared: 0.1174, Adjusted R-squared: 0.09923 F-statistic: 6.453 on 2 and 97 DF, p-value: 0.002338

Frame C.2

C.2. Working with asymptotic distribution

C.2.1. Asymptotic

Now for our second set of experiments we will study how the asymptotic version of the tests performs and how good the approximations are.

For our first experiment, we will study empirically the convergence of our tests to the asymptotic distribution, or it's approximation. For this purpose we will take bivariate Gaussian with correlation 0 with sample sizes 50, 100, 150 ,200, 500 and 1000, in order to decide how good or bad our approximations are, we will perform a Kolmogorov-Smirnoff homogeneity test.

First of all we will start with RDC:

For size 500, we've obtained a pvalue of 0.3564, therefore we accept the null hypothesis H_0 : $RDC \sim \mathcal{X}_9^2$ for significance levels of 0.1,0.05 and 0.01. Figure C.1 shows the pdf, qq plot, pp plot and CDF of our statistic with the one of the $\mathcal{X}_9^2$distribution

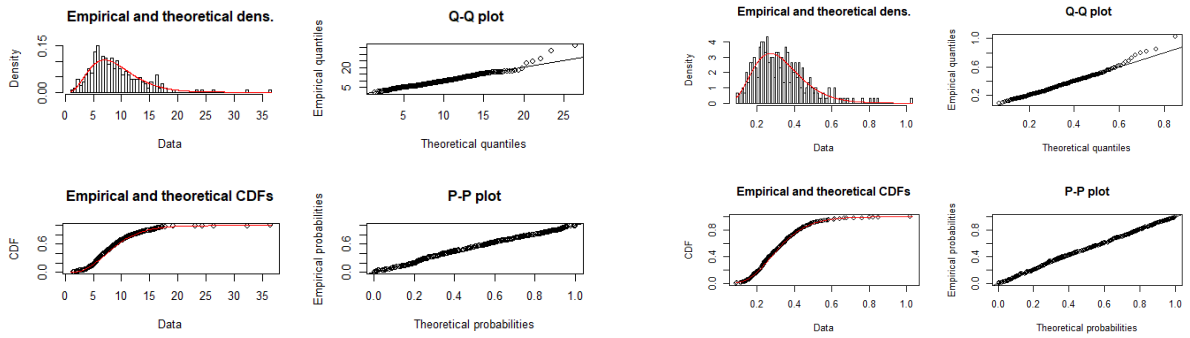
With HSIC distribution:

For size 500, we've obtained a pvalue of 0.1564, therefore we accept the null hypothesis H_0 : $RDC \sim \mathcal{X}_9^2$ for significance levels of 0.1,0.05 and 0.01. Figure C.1 shows the pdf, qq plot, pp plot and cdf of our statistic with the one of the $\mathcal{X}_9^2$ distribution

For an in depth analysis head to the Appendix C, we included the same experiment for different sizes and adding Gaussian noise.

Now that we have accepted our hypothesis we will analyze how good they are. We will compare the power of the asymptotic version with the *real* one on various scenarios.

In our first experiment we will analyze them with a bivariate Gaussian with sizes 50, 100, 150, 200,



(a) RDC statistic with a chi-squared distribution with 9 degrees of freedom

(b) HSIC statistic with a gamma distribution

Figure C.1: Asymptotic comparison of our statistics

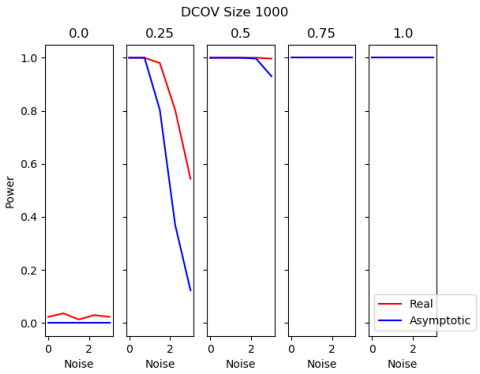
500 and 1000, with different correlations 0, 0.25, 0.5, 0.75, 1.

$$X, Y \sim \mathcal{N}(0, \Sigma_i)$$

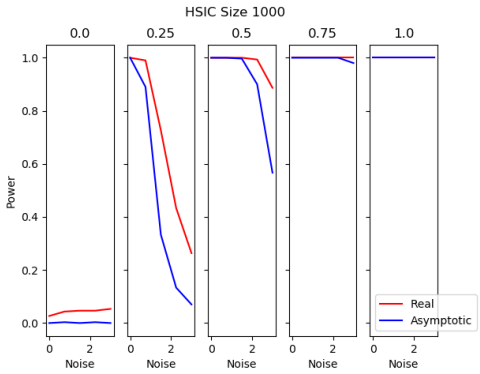
$$\Sigma_1 = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \Sigma_2 = \begin{vmatrix} 1 & 0{,}25 \\ 0{,}25 & 1 \end{vmatrix} \Sigma_3 = \begin{vmatrix} 1 & 0{,}5 \\ 0{,}5 & 1 \end{vmatrix} \Sigma_4 = \begin{vmatrix} 1 & 0{,}75 \\ 0{,}75 & 1 \end{vmatrix} \Sigma_5 = \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix}$$

To the Y variable we will add Gaussian noise going from 0 to 3 $Y = Y + \mathcal{N}(0, \text{noise})$.

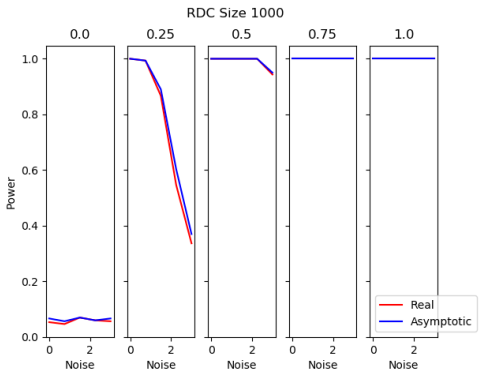
Figures C.2(a) , C.2(c), C.2(b), showcase the power of the real tests vs the asymptotic version for sample sizes of 1000 for DCOV,RDC and HSIC respectively. In the Appendix C it's show this experiment for sample sizes 50,100,150,200 and 500.



(a) Power comparison between the asymptotic and the real version of DCOV for sample size 1000



(b) Power comparison between the asymptotic and the real version of HSIC for sample size 1000



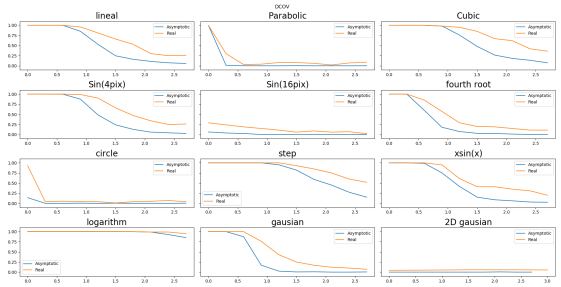
(c) Power comparison between the asymptotic and the real version of RDC for sample size 1000

Figure C.2: Power comparison between the asymptotic and the real version of DCOV for sample size 1000

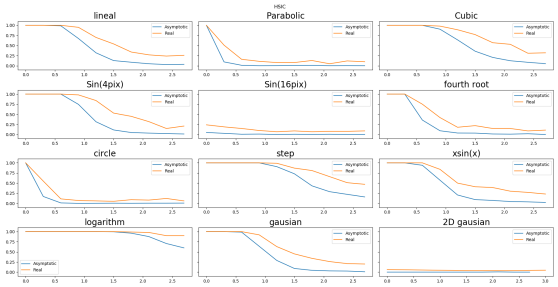
For all tests we've seen how our null hypothesis is always conservative, this is always useful for situations where computation time is critical and the asymptotic version may be better because it minimizes type 1 error.

Now we will study the differences one can see in the previous experiments if we perform the test with the asymptotic version instead of the *real* one.

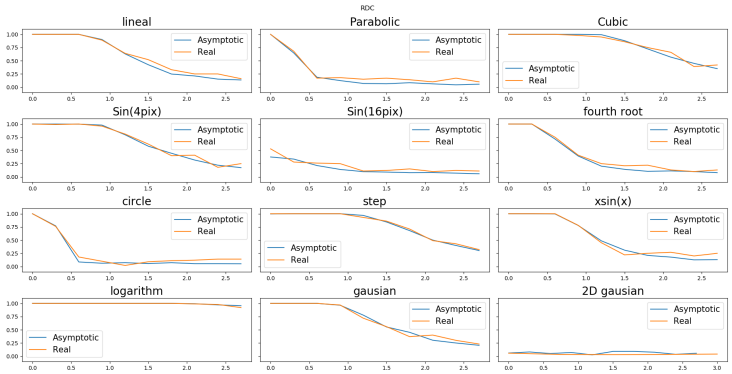
Starting with the first experiment 5.1, we reproduced the same experiment, with a significance level of 0.05, sample size of 200 and Gaussian noise going from 0 to 3. Figures C.3(a), C.3(b), C.3(c) show the asymptotic behaviour of DCOV, HSIC and RDC against the original one respectively for the relation patterns shown in Figure 5.1



(a) DCOV



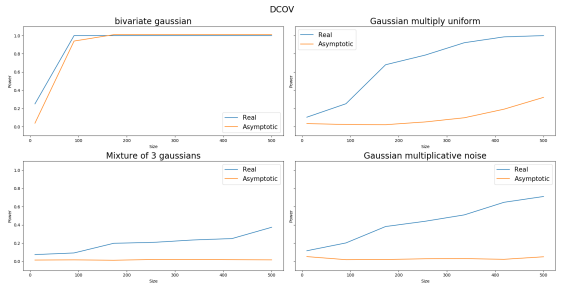
(b) HSIC



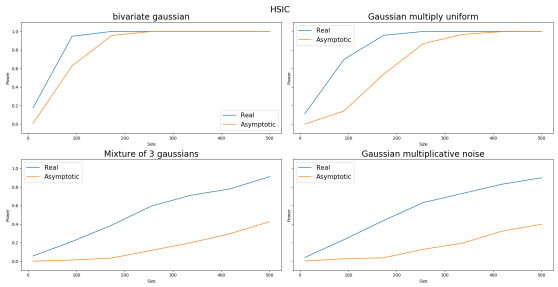
(c) RDC

Figure C.3: Power comparison between the asymptotic and the real version of DCOV,HSIC and RDC for different relation patterns with sample sizes of 200, significance level of 0.05 and Gaussian noise from 0 to 3

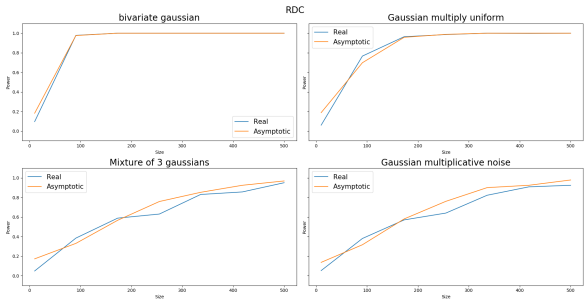
In the second experiment where we studied how good our test was for different sizes and , in this experiment we will see how RDC will outperform the other two tests in their asymptotic behaviour, this may be explained by the fact that HSIC and DCOV asymptotics distributions used for the test were good approximations of the real one, while in RDC we used the actual asymptotic distribution. Figures C.4(a), C.4(b), C.4(c) show respectively the obtained results.



(a) DCOV



(b) HSIC



(c) RDC

Figure C.4: Power comparison between the asymptotic and the real version of DCOV,HSIC and RDC for different relation patterns with sample sizes of varying from 10 to 500, significance level of 0.05

For our last experiment we studied how rotating variables may affect the power of our tests, Figure 5.5 shows samples of the same data with different rotation angle. Figure C.5 presents shows that the asymptotic version of our tests is more conservative than the experimental one. This behaviour was already seen in all the previous experiments.

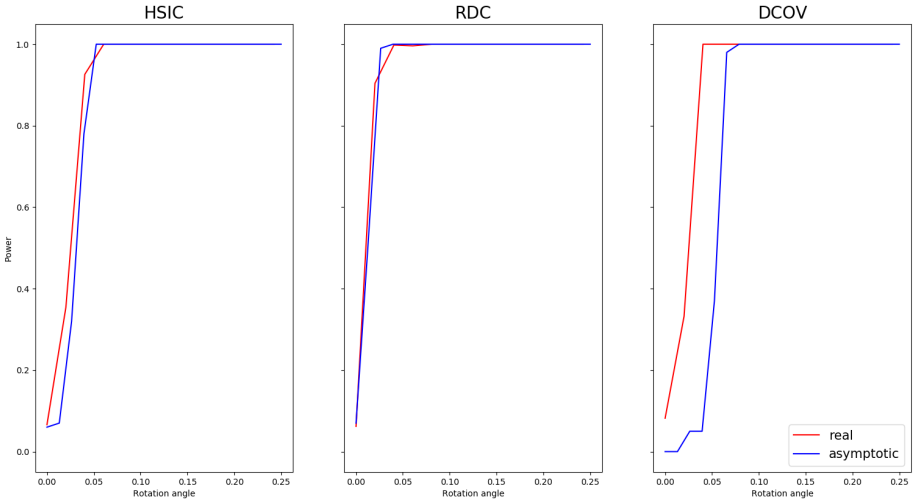


Figure C.5: Power comparison between the asymptotic and the real version of HSIC,RDC and DCOV for different rotation angles of the same data

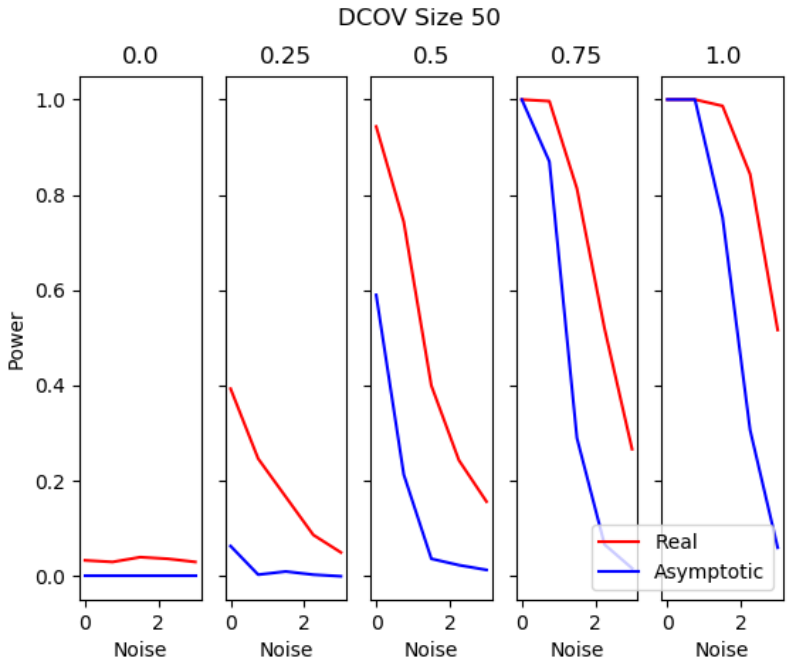


Figure C.6: Power comparison between the asymptotic and the real version of DCOV for sample size 50

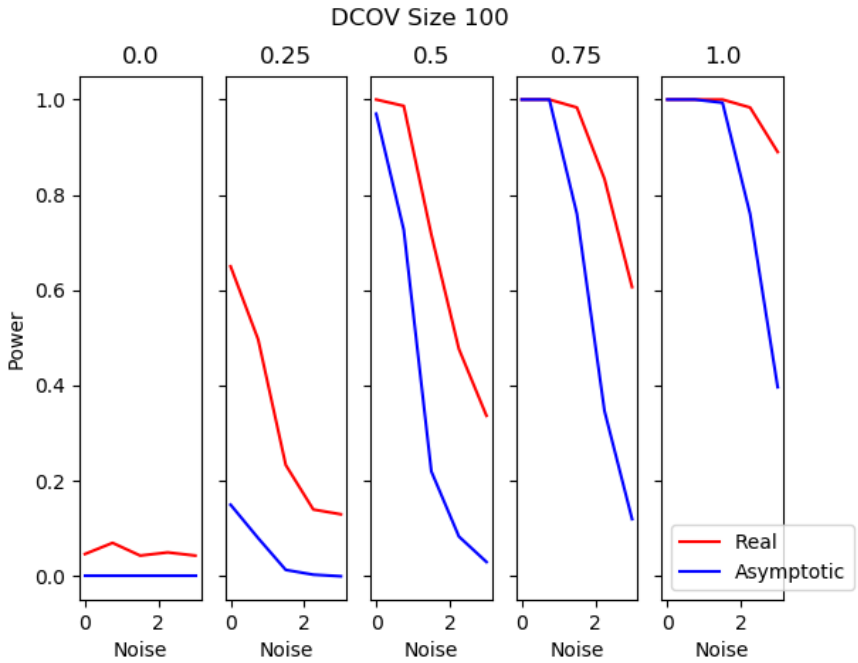


Figure C.7: Power comparison between the asymptotic and the real version of DCOV for sample size 100

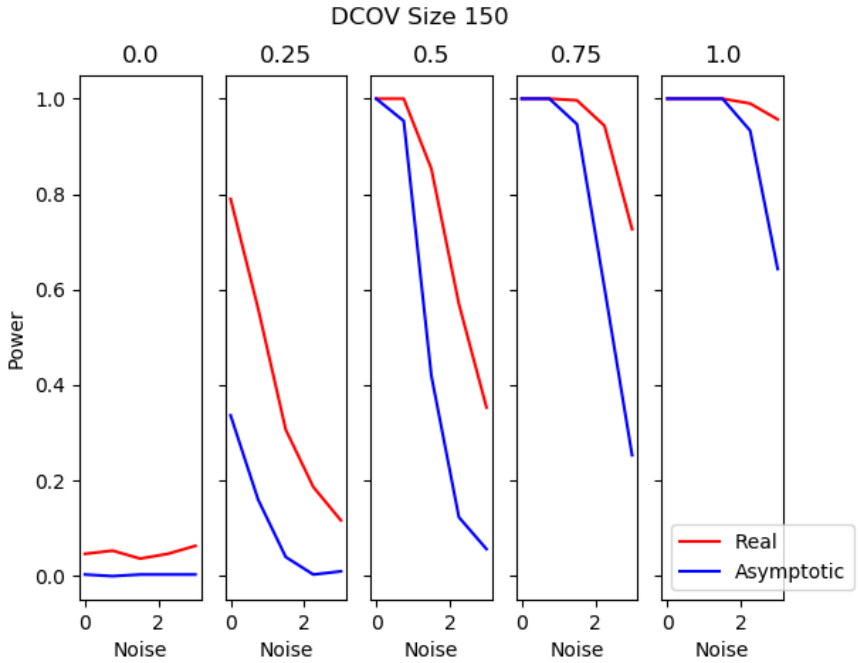


Figure C.8: Power comparison between the asymptotic and the real version of DCOV for sample size 150

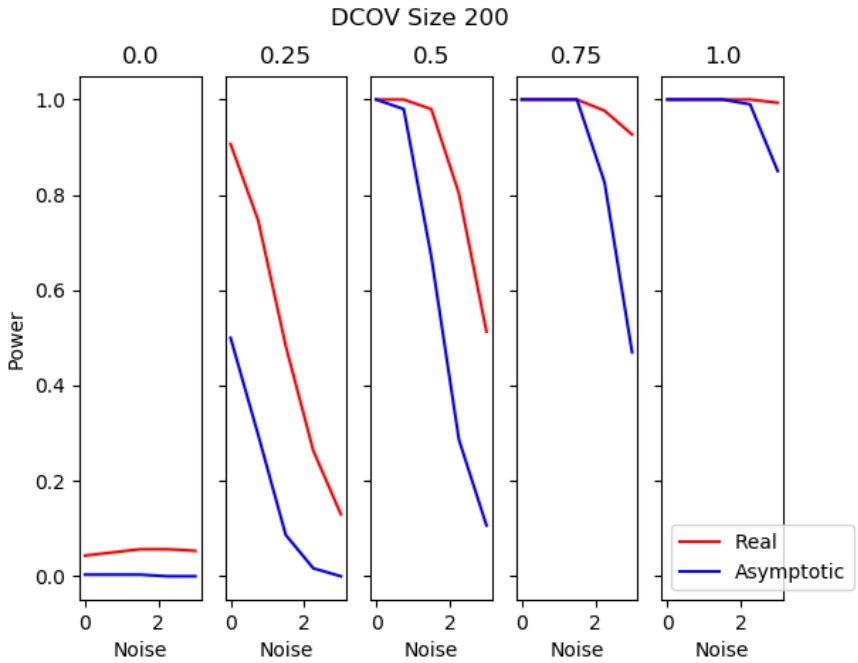


Figure C.9: Power comparison between the asymptotic and the real version of DCOV for sample size 200

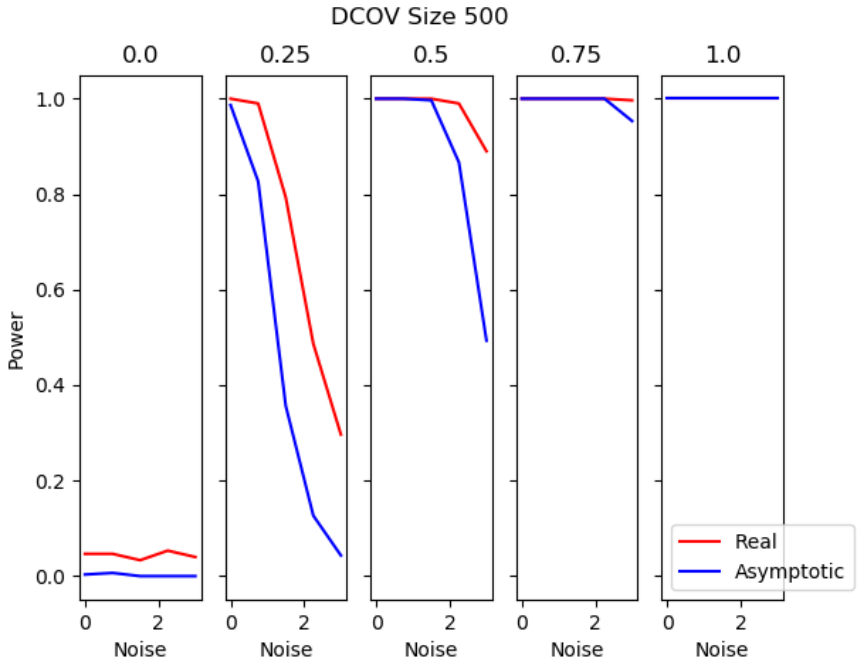


Figure C.10: Power comparison between the asymptotic and the real version of DCOV for sample size 500

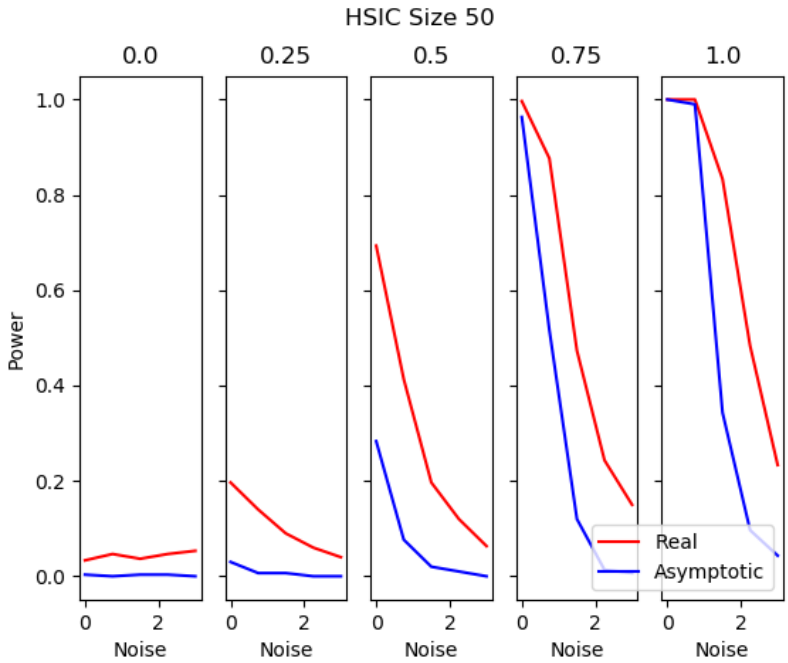


Figure C.11: Power comparison between the asymptotic and the real version of HSIC for sample size 50

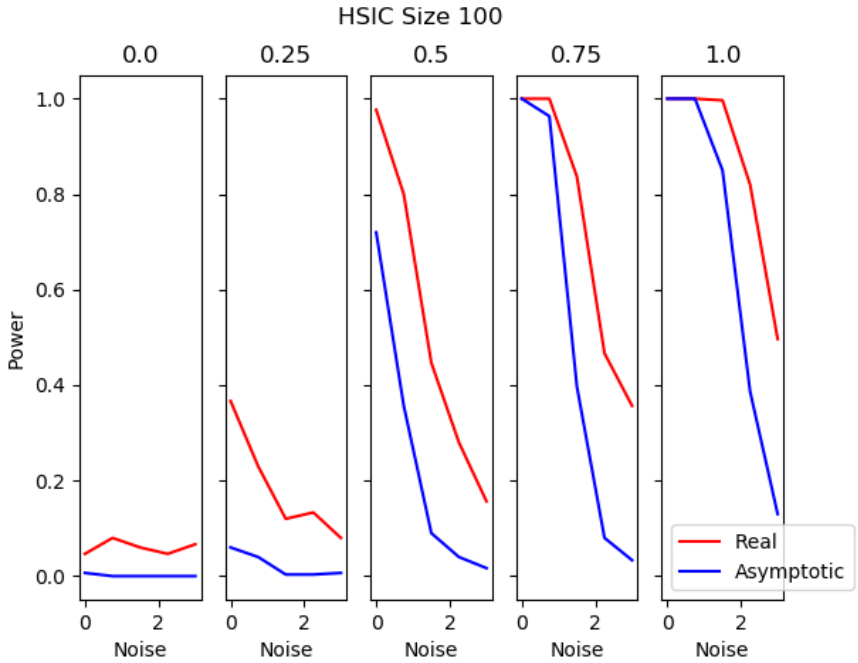


Figure C.12: Power comparison between the asymptotic and the real version of HSIC for sample size 100

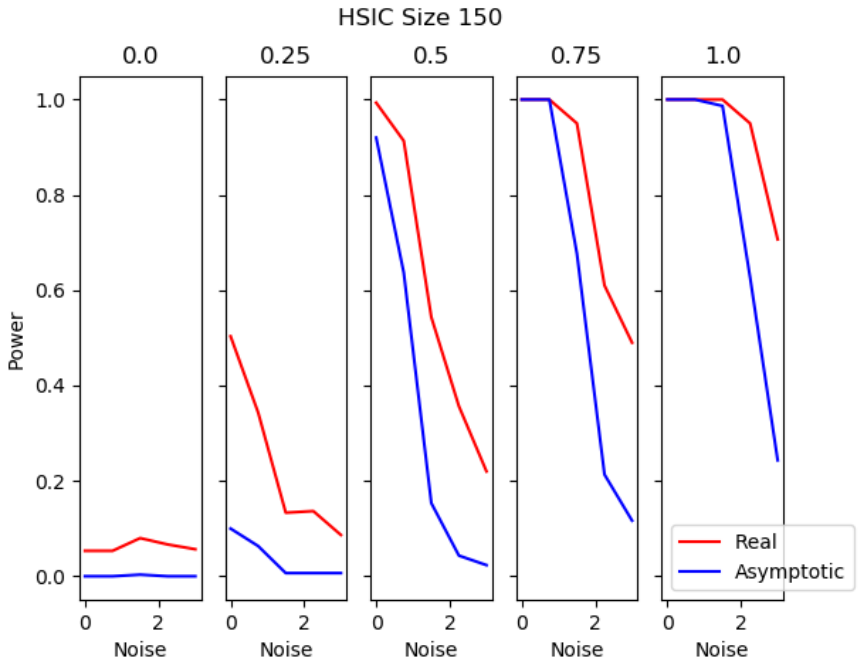


Figure C.13: Power comparison between the asymptotic and the real version of HSIC for sample size 150

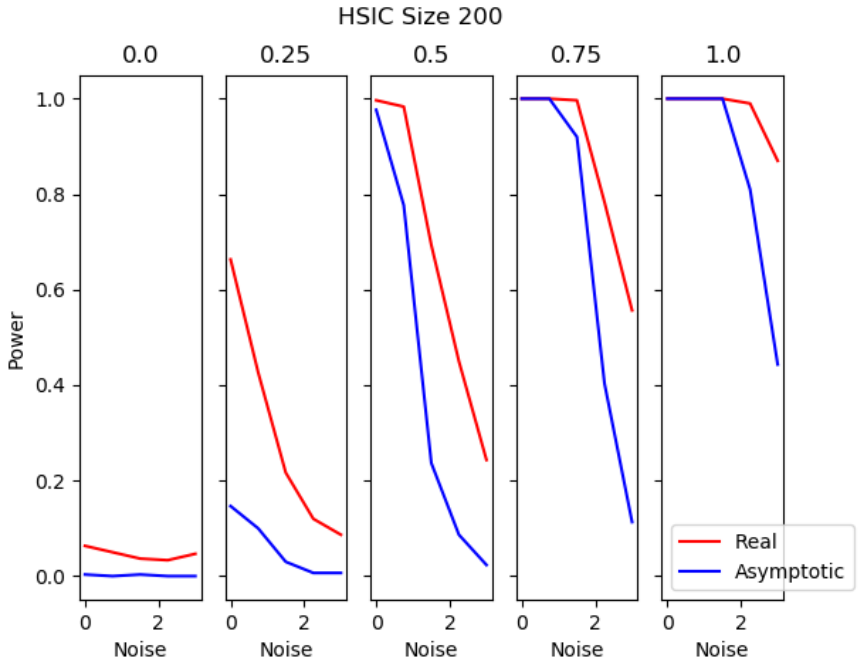


Figure C.14: Power comparison between the asymptotic and the real version of HSIC for sample size 200

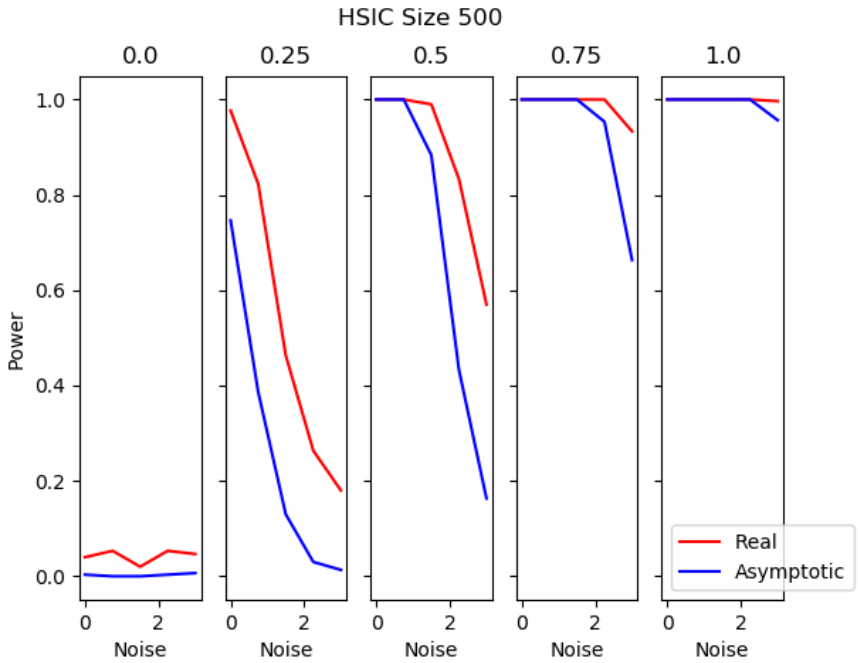


Figure C.15: Power comparison between the asymptotic and the real version of HSIC for sample size 500

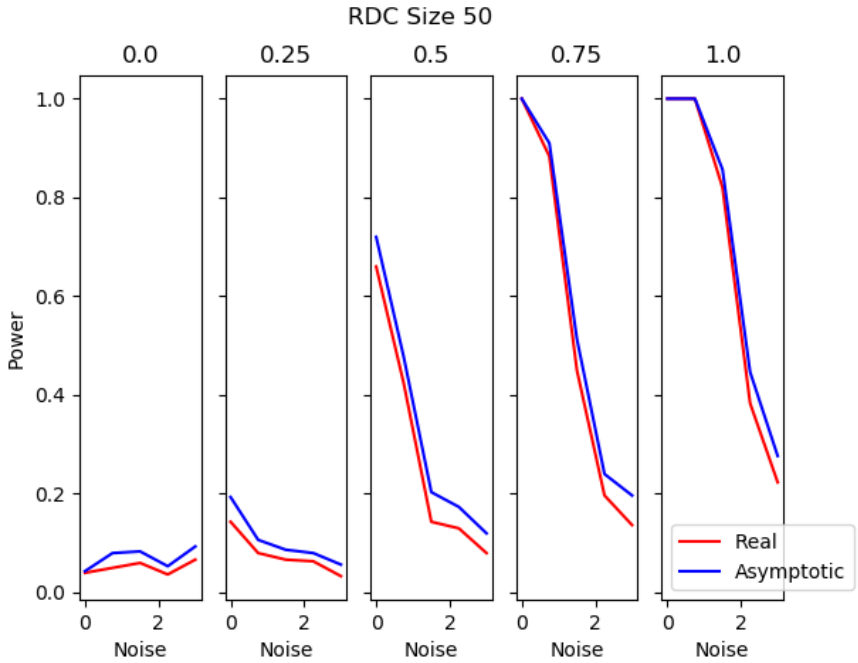


Figure C.16: Power comparison between the asymptotic and the real version of RDC for sample size 50

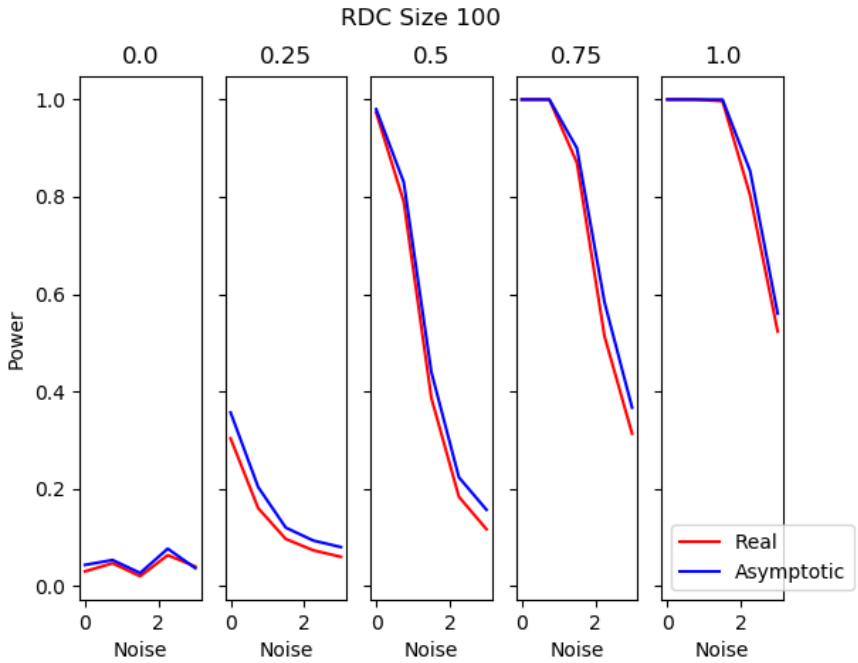


Figure C.17: Power comparison between the asymptotic and the real version of RDC for sample size 100

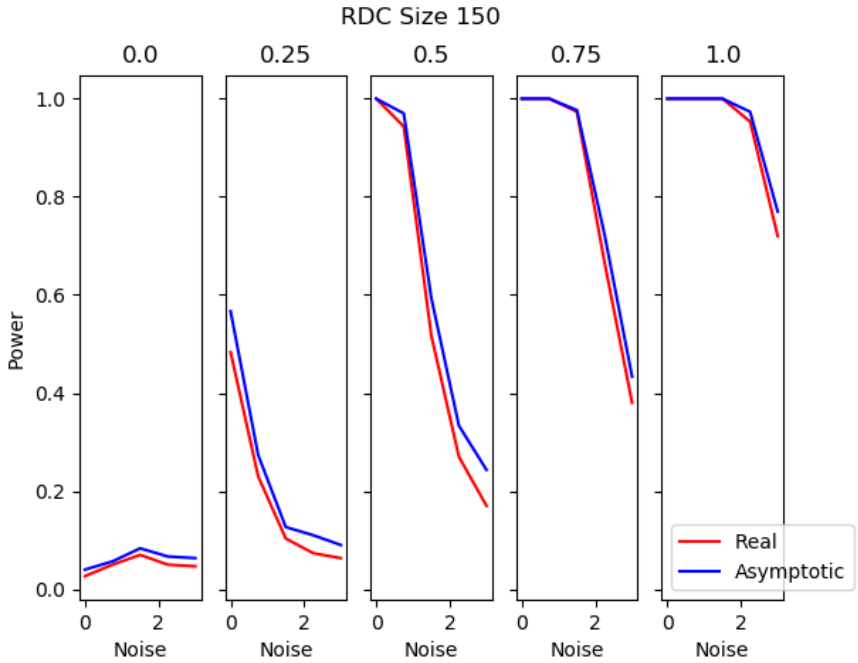


Figure C.18: Power comparison between the asymptotic and the real version of RDC for sample size 150

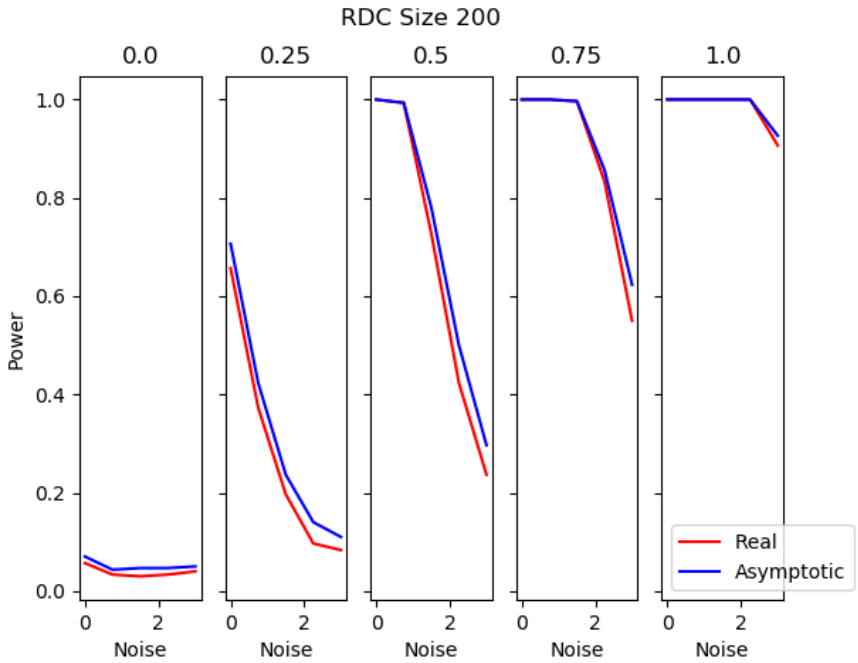


Figure C.19: Power comparison between the asymptotic and the real version of RDC for sample size 200

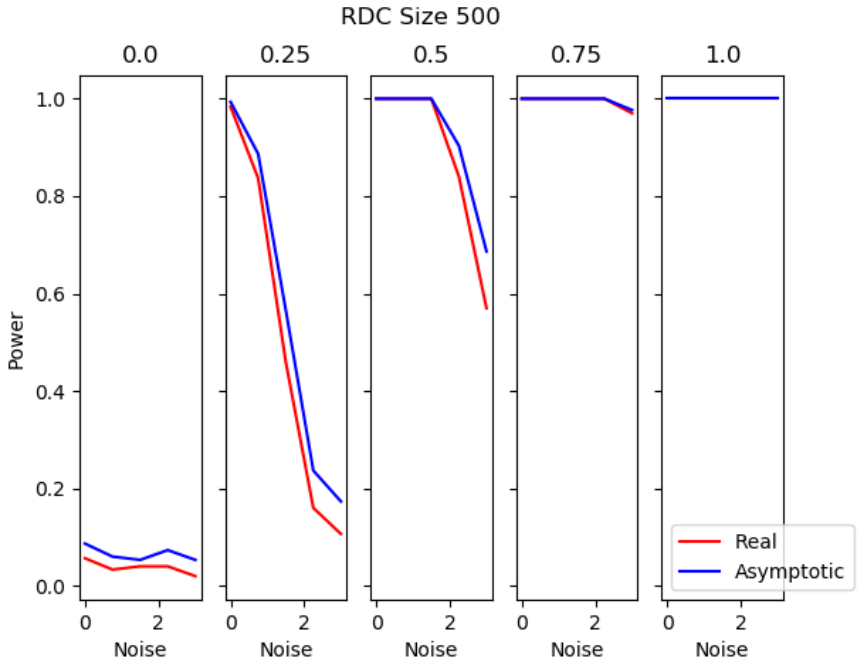


Figure C.20: Power comparison between the asymptotic and the real version of RDC for sample size 500